

# LAYERED GRAPH LOGIC AS AN ASSERTION LANGUAGE FOR ACCESS CONTROL POLICY MODELS

MATTHEW COLLINSON, KEVIN MCDONALD, AND DAVID PYM

ABSTRACT. We describe a uniform logical framework, based on a bunched logic that combines classical additives and very weak multiplicatives, for reasoning compositionally about access control policy models. We show how our approach takes account of the underlying system architecture, and so provides a way to identify and reason about how vulnerabilities may arise (and be removed) as a result of the architecture of the system. We consider, using frame rules, how local properties of access control policies are maintained as the system architecture evolves.

## 1. INTRODUCTION

Access control policies are primarily concerned with controlling information flow within systems. Such control is achieved primarily by restricting — according to the clearance levels of principals (or agents), the classifications of data items, and the locations of data items — the extent to which agents can manipulate (e.g., read, write, copy, move) data.

In formulating models of access control policies, it is commonplace to consider clearances and classifications of agents and data that are based on the idea of (lattices of) multiple labels, multiple levels, and multiple regions. For example, the Bell-LaPadula (BLP) model [5] is intended to protect the confidentiality of data by ensuring that an agent at a given level may not read data that is located at a higher level and may not write data to locations at lower levels; that is, agents may only create data at or above their own level of confidentiality. Similarly, the Biba model [7] is intended to protect the integrity of data by ensuring that a agent at a given level may not read data that is located at a lower level and may not write data to a higher level: that is, agents may only create data at or below their own integrity level. These restrictions, which constitute the *Strict Policy* of the Biba model, will be referred to as the Biba model throughout. Brewer and Nash’s Chinese Walls model [11], in contrast, is concerned with multiple, disjoint, non-overlapping, regions. It is intended to protect the confidentiality of data by ensuring that agents are not subject to conflicts of interest: that is, an agent may read only data that is located in a region that is not in conflict with the agent’s own region. Also important is the stateful Clark-Wilson model [16]. In this model, certain groups of agents are permitted to modify certain types data items, but in some instances multiple authorizations are required, so that at any one time a modification may be partially authorized, and audit trails must be maintained.

Systems for which access control policy models are formulated are typically complex assemblies of people, processes, and technology embedded within complex logical and/or physical architectures. The hierarchical structure of access control policies must be implemented for such systems which, typically, carry a rich structure of layers and regions. In this paper, we employ the graph-theoretic model of layered architectures introduced by the present authors in [21] to provide a simple yet flexible model of the the underlying system architecture that is a model of a sub-structural logic that captures the concept of layering using some very weak (non-commutative, non-associative) multiplicative connectives. Access control policies, and indeed security policies and architectures more generally, are defined on top of such an architecture and describe the constraints with which agents are expected to comply when interacting with that architecture. In this paper, we introduce a specific instance of the logic of layered architectures that allows access control policies for specific architectures to be represented. Such a set-up supports an understanding of how a policy must be adapted to the specific system architecture to which it is intended to apply and, indeed, how mismatches between the formulation of a policy and the system architecture may lead to security vulnerabilities. An example is discussed below.

In modelling the interaction between system architecture and security policies, we make a simplifying assumption: that the layers in the system architecture correspond to the levels in the access control policy model. For the examples we explore in the sequel, this assumption neither trivializes the models nor introduces unnecessary confusion. Rather, it allows us to focus directly, with minimal coding, on how the connectivity between layers either supports or undermines the access control policy.

The layered graphs considered in [21] provide a concrete semantics for a substructural logic, **LGL**, of layered graphs. We show how **LGL** can be used to describe access control policy models in the context of hierarchically structured (or layered) systems, as modelled in [21]. **LGL** combines, in a manner similar to BI [45] and Separation Logic [52], classical propositional additives and multiplicatives that, in our case, are assumed to be neither commutative or associative. These very weak multiplicatives describe the layering structure of graphs; that is, how one graph may be layered over another, and so on.

The logic **LGL** is related to the system DW considered by Read [51], in which the multiplicative (intensional) conjunction (corresponding to the multiplicative bunching operation) is commutative and in which the multiplicative (intensional) implications (handed versions of linear implication, as in Linear Logic, BI [45], and Lambek’s systems, such as [42] and earlier papers) that are naturally present in our system are absent. Read considers a hierarchy of extensions of DW, including axioms for associativity and many stronger structural properties. Read’s hierarchy recovers the hierarchy of relevant systems up to R and classical logic itself (e.g., see [51]). Just as in these systems, we work with a classical negation, deferring consideration of weaker systems of negation to another occasion. Layering need not be defined in one direction only: it may be that two graphs are layered over each other. In modelling terms, this would mean that, while it is useful to separate the two layers, resources can flow both up and down. To this end, **LGL** includes a notion of ‘bi-layering’, that is consistent with our basic notion of layering. **LGL** also considers a class of layered graphs that amount to the intuitive notion of a stack.

For an example, to help clarify the basic ideas, consider an organization that assigns the clearance levels secure and general to agents and data items, partitioning its network accordingly into Secure and General layers. It is the intention of the organization that no data should pass from the secure to general level. The organization could choose to maintain this segregation of data by enforcing an access policy preventing an agent that is in the Secure Layer from writing downwards into the General Layer. However, this approach is inadequate as the organization is not taking *all* of the underlying structure into account.

Now consider Figure 1. This shows the General and Secure Layers connected by a Mobile Stor-

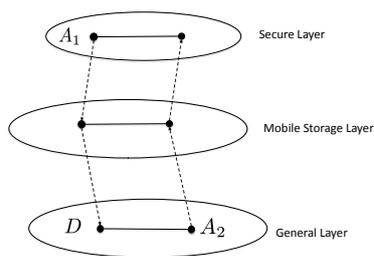


FIGURE 1. Organizational Structure including Mobile Storage Layer

age Layer, giving a much truer illustration of the underlying structure of the organization (which, indeed, is the structure of many organizations today). The widespread usage of devices such as USB memory sticks can introduce side-channels, meaning that any organization must consider the impact this may have on access to data within its system. The policy of the organization does not prevent data from being written from the Secure Layer to the Mobile Storage Layer and then into the General Layer. This means that the policy put in place by the organization has been violated as the underlying structure was not properly considered. Note that in Figure 1, and in all further figures, we draw explicitly only the vertices and edges that are relevant to the situation being described and dotted lines are used to indicate an edge that connects one (sub)graph to another.

In all such examples, it is important to remember that, as with all forms of mathematical modelling, the analysis that is possible depends critically on the quality of the model in respect of its intended use and cannot address aspects of the system that are not represented in the model [18]: ‘the map is not the territory’ [40].

As another example, consider the construction of a system to support the rules of both the BLP and Biba models. When implementing multiple policies, the system architecture must not cause one policy to fail in the presence of the other. This is illustrated in Figure 2, where Agent

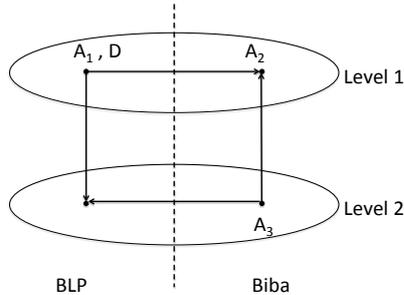


FIGURE 2. Location of Agents and Data

$A_1$  can write  $D$  to level 2 on the right-hand system. Moreover, if  $D$  can reach the location of  $A_3$  then it could be written back across to level 2 of the left-hand system. The construction of the system must therefore be analysed to ensure full compliance with all appropriate rules.

Finally, consider an organization using the Chinese Walls model to segregate its physical network and prevent agents from accessing data items that are deemed to conflict. Suppose that the organization also employs outside consultants that are given mobile access to data through a laptop provided by the organization. This usage of laptops causes an additional layer of organizational structure, the Mobile Access Layer, which introduces a side-channel in a similar way to that described earlier. Since the rules of the Chinese Walls model do not guard against information flow through additional layers of a network, the intended policy of segregation can be violated. Consider Figure 3. This shows that agent  $A_3$  could potentially read data item  $D_1$  and subsequently

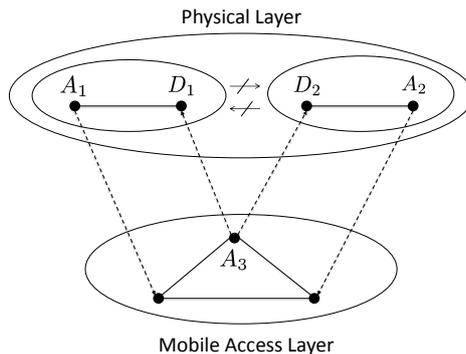


FIGURE 3. Chinese Wall in a Layered Graph

write a copy upwards that could then be read by  $A_2$ , which should not be permitted. In order to preserve the intended segregation, the organization must implement additional rules to guard against the type of multi-level access described.

## 2. RELATED WORK AND CONTRIBUTION

The work we present here has little preceding work in the literature. The usual approach to understanding access control policy models — see [3] and [4] for efficient summaries — is to model

only security architecture of system, such as the levels of classification of information in multi-layer models or the separated regions in multi-lateral models, and the roles of the agents to which the policy is intended to apply. Historically, the technique for representing multi-level access control policies has been to use *lattice models*, [24, 53]. This approach, while often effective because the rules of multi-level policies rarely give consideration to the underlying system architecture, does not offer adequate scope to investigate the composition of multiple models. It is also rare for multi-level policies to be studied in conjunction with multi-lateral policies such as Chinese Walls. An exception can be found in [47], where restrictions are placed on a pre-existing lattice model to incorporate both clearance level and a notion of conflict of interest. A useful early discussion of policies and models is in [31].

Our approach differs quite explicitly in that we model not only the security architecture of the system, and the interactions of the agents with it, but also the underlying system architecture. Our modelling approach is based in mathematical logic and, in particular, uses models of a particular system of logic. The basic idea is quite simple. The propositions of the logic, expressed in the language of the logic, are used to express access control policies in the sense described above. This part of our work sits in the tradition represented, for example, by papers such as [2], [43], and [1]. Models of the logic based on layered graphs are used to represent the system architecture to which the policies are intended to apply. The closest work in the literature to this is represented by the ideas presented in [37] and [52], describing Pointer Logic and Separation Logic, respectively. In these papers, the propositions of the logic — which, like **LGL**, is also based on BI [45, 50, 29] — are used to express the properties of mutable data structures, such as pointers, during the execution of (imperative) programs and models of the logic are built using the stack and heap of the (abstract) machine that executes the programs.

Our use of layered graphs here represents an abstraction and simplification of the approach to representing distributed system structure presented in [19, 18] in which a process calculus (LSCR) of locations, resources, and processes, together with an associated modal logic (LMBI) is set up. The calculus LSCR has an operation judgement of the form  $L, R, E \xrightarrow{a} L', R', E'$ , asserting that process  $E$  evolves by action  $a$ , relative to the available resources  $R$  at location  $L$ , to become the process  $E'$  that may further evolve relative to resources  $R'$  at location  $L'$ . In this set-up, the algebra of processes is a synchronous one that builds on SCCS to incorporate resources and locations [18]. Resources are modelled essentially as in the bunched logic BI [45, 50, 29] and its derivative Separation Logic [52], capturing basic axioms of combination and comparison of resource elements. Locations are modelled [18, 17] by structures, such as directed graphs, hyper-graphs, and certain topological spaces, that satisfy some basic axioms — that there are directed connections between places, that there is a notion of sub-location and a notion of substitution, supporting refinement and abstraction, and, possibly, a product of locations.

Associated with the calculus is the Hennessy–Milner-style modal logic MBI, with satisfaction judgement  $L, R, E \models \phi$ , asserting that, relative to resources  $R$  at location  $L$ , process  $E$  has property  $\phi$ . The value of this approach in studying access control has been demonstrated in [19, 18] in which the conjecture of Abadi et al. [2] — that for one agent to act in the role of another is a form of concurrent composition — is resolved positively.

Belnap logic has been used in [12] to study the composition of access control policies through the use of a four-valued predicate where judgements are based on the level of information contained in a piece of data and the trustworthiness of data items. There has also been some efforts in [10, 15] to compose policies. The work in these papers relies on policies already having a pre-existing logical framework and an associated interpretation function. Set operations, for example union and complementation, are then applied to the images of the interpretation functions in order to describe composition. Other logics have also been used to specify access control models. For example, the modal logic S5 is used in [22] and in [30] a model of *relation-based access control* is given through the use of description logic. Finally, [41] presents an extension of [2, 43] in order to represent Role-Based Access Control (RBAC).

Access control policies have been specified using so-called *system graphs* and *policy graphs* in [35]. System graphs are created using labelled vertices to describe users, or files, names and clearance levels while directed edges are labelled with actions. Policy graphs represent access

restrictions and are applied to the system graphs by matching vertex labels. This programming-based approach, though clearly related in spirit to our view, does not provide clear accounts of the system architecture and security policy models and does not provide mathematical tools for reasoning about compliance or composition.

Access control has also been studied using category theory in [6] to build what the authors describe as a *meta-model*, which they use to represent various access control policies and compose, for example, RBAC and BLP. In [39], a category is formed using graphs and partial graph morphisms. Access control rules are then given by injective morphisms, leading to the definition of a framework for giving access control policies. While having some connections in spirit to our approach, particularly in their focus on composition, these approaches do not provide tools for reasoning about how systems comply with policies.

In our **LGL**-based approach, compositionality is available in all aspects of system models and their associated policies. First, logical expressions of policy, built from a class of atomic access operations, introduced in Section 4, can be composed using the full strength of **LGL**: classical additives, non-commutative layering multiplicatives, and, in a later section, BI's multiplicative conjunction ( $*$ ) and implication ( $-*$ ). Second, the underlying system architecture can be composed by layers, building up a hierarchical structure. Third, additional structural complexity within and between layers can be added by composing graphs in the usual ways. These aspects of compositionality are illustrated extensively in the examples used throughout the paper.

In Section 3, we briefly review the logic of layered graphs **LGL**, introduced with substantive theoretical development in [21], summarize its meta-theory and present an  $n$ -ary composition that can be used to represent a stack. In Section 4, we introduce a specific model of **LGL** that captures the core notion of access control in the setting of layered graphs. We show how this set-up can be used to model the implementation of access control policy models (such as Bell-LaPadula and Biba) in the context of hierarchically structured systems, and show how various vulnerabilities may be understood. In Section 5, we consider how the usual multiplicatives, familiar from BI and Separation Logic, can be added and used, within layers, to capture access control policies described by Chinese Walls. In Section 6, we consider an extension of the logic with actions and modalities that describe how the underlying system architecture may evolve and consider how naturally occurring frame rules, both within and between layers, can be used to express invariant properties. of access control policies. These latter two sections describe rich forms of compositional structure that are present in our set-up.

We conclude, in Section 7, with some brief remarks on a few technical directions for the specific work of this paper and, more importantly, a substantial discussion of the broader scientific context for the technical work presented here. In particular, we discuss how the work presented in this paper contributes to an overall project to bring to bear logical and economic modelling to analyse security, design, behaviour, and decision-making, with the aim of delivering a methodology to support the design and implementation of systems and policies for access control that takes appropriate account not only of security policy objectives but also of the architecture of the system to which policies apply and the behaviour of agents with the context the both the policy and the system architecture.

### 3. LAYERED GRAPH LOGIC

This section has two parts. First, we review very briefly the logic, **LGL**, of layered graphs as introduced in detail in [21]. Second, we present an  $n$ -ary composition that is used to identify a particular class of layered graphs called *bi-layered stacks*. The logic **LGL** is intended to capture the concept of layering that is commonly found in the complex systems literature, as discussed in [21]. Conceptually, our point of departure is the substructural (bunched) modal process logic, LMBI, described in [18, 19, 17], associated with the calculus LSCR<sub>P</sub> of locations — where the concept of layering resides — resources, and processes [18, 19, 17]. This framework is used, in [20], to resolve positively a conjecture in [2] that one agent acting in the role of another is a form of concurrent composition.

**3.1. Layered Graph Logic.** The layered graph logic, in its basic form, focusses on worlds that are locations and, in particular, directed graphs. Standard graph-theoretic notions are used throughout this paper. All graphs are assumed to be directed. For any graph  $G$ , let  $V(G)$  be the set of

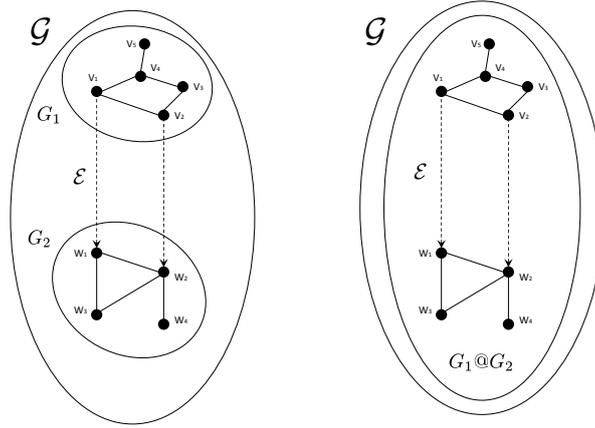


FIGURE 4. Example of Layer Construction

vertices of  $G$ ,  $E(G)$  be the set of edges of  $G$ , and  $\text{Sg}(G)$  be the set of subgraphs of  $G$ . The notation  $H \sqsubseteq G$  means  $H \in \text{Sg}(G)$ .

We are often interested in situations in which we are given a graph  $\mathcal{G}$  and a distinguished set of edges  $\mathcal{E}$  of  $\mathcal{G}$ , and we then consider properties of the set of subgraphs of  $\mathcal{G}$ . In such a situation, we often refer to  $\mathcal{G}$  as the *ambient graph* and  $\mathcal{E}$  as the *distinguished edge set*. Let  $G_1$  and  $G_2$  be any two subgraphs of  $\mathcal{G}$ . The notation  $G_1 \rightsquigarrow_{\mathcal{E}} G_2$  is used to signify the fact that  $G_2$  is reachable from  $G_1$  via an edge of  $\mathcal{E}$ . The notation  $\not\rightsquigarrow_{\mathcal{E}}$  is the complement of the relation  $\rightsquigarrow_{\mathcal{E}}$ . Let  $G_1$  and  $G_2$  be any two such subgraphs. Write  $\mathcal{G}(G_1, G_2)$  for the set of edges of the ambient graph that connect any vertex of  $G_1$  to any vertex of  $G_2$ . Define  $\mathcal{G}[G_1, G_2] = \mathcal{G}(G_1, G_2) \cup \mathcal{G}(G_2, G_1)$ . We make use of partial definedness throughout: the notation  $X \downarrow$  means that a given expression  $X$  is defined.

Given a graph  $\mathcal{G}$  and a distinguished set of edges  $\mathcal{E}$ , we define a partial, binary operation  $@ : \text{Sg}(\mathcal{G}) \times \text{Sg}(\mathcal{G}) \rightarrow \text{Sg}(\mathcal{G})$  as follows. For any subgraphs  $G_1$  and  $G_2$  of  $\mathcal{G}$ , the value  $G_1 @ G_2$  is defined if and only if  $V(G_1) \cap V(G_2) = \emptyset$  and  $G_1 \rightsquigarrow_{\mathcal{E}} G_2$  and  $G_2 \not\rightsquigarrow_{\mathcal{E}} G_1$ . When defined,  $G_1 @ G_2$  is given by  $V(G_1 @ G_2) = V(G_1) \cup V(G_2)$  and  $E(G_1 @ G_2) = E(G_1) \cup E(G_2) \cup (\mathcal{E} \cap \mathcal{G}[G_1, G_2])$ . When  $G_1 @ G_2$  is defined, it gives the (disjoint) union of the graph arguments, together with the edges of  $\mathcal{E}$  between  $G_1$  and  $G_2$ . The definition of this operator is relative to the given ambient graph  $\mathcal{G}$  and distinguished set of edges  $\mathcal{E}$ . Where the distinguished set of edges  $\mathcal{E}$  needs to be emphasized we write  $@_{\mathcal{E}}$ . We say that a graph  $G$  is *layered* if  $G = G_1 @ G_2$  for some  $G_1, G_2$ . An example of this composition, which is non-commutative and non-associative, is shown in Figure 4.

We define a logical language, **LGL**, for expressing layering properties of graphs. The logic combines classical additives and a non-commutative, non-associative multiplicative conjunction (and its adjoints) in the style of BI [45, 50]. The logic is described in detail in [21]. A different logic for graphs, with a separating (associative and commutative) conjunction is explored in [13, 23].

We give an interpretation of **LGL** that uses the layered graph composition described above. The interpretation requires a little extra structure in the layered graphs, which we call a *scaffold*. We say that a pair  $(\mathcal{G}, \mathcal{E})$  is a scaffold if it has at least one pair of subgraphs  $G_1, G_2$  with  $G_1 @ G_2$  defined.

Assume a set  $\text{Atoms}$  of atomic propositions, ranged over by  $p$ . The set,  $\text{Formulae}$ , of all propositional formulae is generated by the following grammar:

$$\phi ::= p \mid \top \mid \perp \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \rightarrow \phi \mid \phi \blacktriangleright \phi \mid \phi \multimap \phi \mid \phi \blacktriangleleft \phi .$$

The connectives above are the standard classical additives, together with multiplicative conjunction,  $\blacktriangleright$ , and implications  $\multimap$ ,  $\blacktriangleleft$ . We define  $\neg\phi$  as  $\phi \rightarrow \perp$ . A Hilbert-type proof system, **LGL<sub>H</sub>**, for **LGL** is given in Table 1. The system combines a Hilbert-type system for the Boolean fragment of **LGL** together with the necessary multiplicative rules for  $\blacktriangleright$ ,  $\multimap$ , and  $\blacktriangleleft$ . These latter rules amount to the evident non-commutative variation of the multiplicative rules in the Hilbert-type system for BI [50]. Natural deduction, sequent calculus, and display calculus systems are given in [21].

1. $\phi \vdash \phi$ 3. $\perp \vdash \phi$ 5. $\frac{\eta \vdash \phi \quad \eta \vdash \psi}{\eta \vdash \phi \wedge \psi}$ 7. $\frac{\eta \vdash \psi \quad \phi \vdash \psi}{\eta \vee \phi \vdash \psi}$ 9. $\frac{\eta \wedge \phi \vdash \psi}{\eta \vdash \phi \rightarrow \psi}$ 11. $\frac{\phi \vdash \psi}{\eta \wedge \phi \vdash \psi}$ 13. $\frac{\eta \blacktriangleright \phi \vdash \psi}{\eta \vdash \phi \blacktriangleright \psi}$ 15. $\frac{\eta \blacktriangleright \phi \vdash \psi}{\phi \vdash \eta \blacktriangleright \psi}$	2. $\phi \vdash \top$ 4. $(\phi \rightarrow \perp) \rightarrow \perp \vdash \phi$ 6. $\frac{\phi \vdash \psi_1 \wedge \psi_2}{\phi \vdash \psi_i} \quad (i = 1, 2)$ 8. $\frac{\phi \vdash \psi_i}{\phi \vdash \psi_1 \vee \psi_2} \quad (i = 1, 2)$ 10. $\frac{\eta \vdash \phi \rightarrow \psi \quad \eta \vdash \phi}{\eta \vdash \psi}$ 12. $\frac{\xi \vdash \phi \quad \eta \vdash \psi}{\xi \blacktriangleright \eta \vdash \phi \blacktriangleright \psi}$ 14. $\frac{\xi \vdash \phi \blacktriangleright \psi \quad \eta \vdash \phi}{\xi \blacktriangleright \eta \vdash \psi}$ 16. $\frac{\xi \vdash \phi \blacktriangleright \psi \quad \eta \vdash \phi}{\eta \blacktriangleright \xi \vdash \psi}$
---	---

 TABLE 1. The Hilbert-type System  $\mathbf{LGL}_H$ 

Given a scaffold  $(\mathcal{G}, \mathcal{E})$  and a valuation  $\mathcal{V} : \text{Atoms} \rightarrow \mathcal{P}(\mathbf{Sg}(\mathcal{G}))$ , where  $\mathcal{P}$  is the powerset operator, the language can be given a semantics on the set of subgraphs of  $\mathcal{G}$ . The satisfaction relation is  $\models \subseteq \mathbf{Sg}(\mathcal{G}) \times \text{Formulae}$ . The definition of the satisfaction relation for the multiplicative connectives is given in Table 2. Satisfaction for the additive connectives follows that of classical propositional logic.

$(\mathcal{G}, \mathcal{E}), G \models_{\mathcal{E}} \phi_1 \blacktriangleright \phi_2$	iff	there are $G_1, G_2$ such that $G = G_1 @ G_2$ , and $(\mathcal{G}, \mathcal{E}), G_1 \models_{\mathcal{E}} \phi_1$ and $(\mathcal{G}, \mathcal{E}), G_2 \models_{\mathcal{E}} \phi_2$
$(\mathcal{G}, \mathcal{E}), G \models_{\mathcal{E}} \phi \blacktriangleright \psi$	iff	for all $H$ , $G @ H \downarrow$ and $(\mathcal{G}, \mathcal{E}), H \models_{\mathcal{E}} \phi$ implies $(\mathcal{G}, \mathcal{E}), G @ H \models_{\mathcal{E}} \psi$
$(\mathcal{G}, \mathcal{E}), G \models_{\mathcal{E}} \phi \blacktriangleright \psi$	iff	for all $H$ , $H @ G \downarrow$ and $(\mathcal{G}, \mathcal{E}), H \models_{\mathcal{E}} \phi$ implies $(\mathcal{G}, \mathcal{E}), H @ G \models_{\mathcal{E}} \psi$

 TABLE 2. The Satisfaction Relation for Multiplicative Connectives of  $\mathbf{LGL}$ 

Let  $\llbracket \phi \rrbracket = \{G \mid G \models \phi\}$  for every proposition  $\phi$ . This defines an interpretation function  $\llbracket - \rrbracket : \text{Formulae} \rightarrow \mathcal{P}(\mathbf{Sg}(\mathcal{G}))$ . Again, this is all relative to  $(\mathcal{G}, \mathcal{E})$  and  $\mathcal{V}$ . With this definition, it is easy to check that the following properties hold for all  $\phi, \psi$ , and  $\xi$ :

$$\llbracket \phi \wedge \psi \rrbracket \subseteq \llbracket \xi \rrbracket \quad \text{iff} \quad \llbracket \phi \rrbracket \subseteq \llbracket \psi \rightarrow \xi \rrbracket$$

$$\llbracket \phi \blacktriangleright \psi \rrbracket \subseteq \llbracket \xi \rrbracket \quad \text{iff} \quad \llbracket \phi \rrbracket \subseteq \llbracket \psi \blacktriangleright \xi \rrbracket \quad \text{iff} \quad \llbracket \psi \rrbracket \subseteq \llbracket \phi \blacktriangleright \xi \rrbracket .$$

These relationships underpin the adjointness relations for the implications  $\rightarrow$ ,  $\blacktriangleright$  and  $\blacktriangleright$ .

Fix a scaffold  $(\mathcal{G}, \mathcal{E})$ , and consider an instance of the satisfaction relation of the form  $(\mathcal{G}, \mathcal{E}), G \models_{\mathcal{E}} \phi_1 \blacktriangleright \phi_2$ , where  $G$  is a subgraph of  $\mathcal{G}$ . This means that  $G$  can be decomposed into subgraphs  $G_1$  and  $G_2$ , that is  $G = G_1 @_{\mathcal{E}} G_2$ , such that  $G_1$  satisfies  $\phi_1$  and  $G_2$  satisfies  $\phi_2$ . The asymmetry of the composition operator, with edges from the component  $G_1$  of  $G$  to the component  $G_2$ , means that it is reasonable to regard  $G_1$  as ‘layered over’  $G_2$ .

3.1.1. *Meta-theory.* An algebraic semantics for  $\mathbf{LGL}$  can be established. See [21] where it is also explained that the semantics on graphs given here is special case of this algebraic semantics. The semantics is expressed in terms of layered magmas — adapting terminology from [26, 27] to describe a structure  $(\mathbb{M}, \bullet)$  with a partial binary operation  $\bullet$  on a carrier set  $\mathbb{M}$  — and layered

algebras  $(\mathbb{A}, \wedge, \neg, \top, \blacktriangleright, \rightarrow, \blacktriangleright)$  which combine Boolean operations with a non-commutative binary operation and  $\blacktriangleright$  and its associated right and left adjoints,  $A \rightarrow -$  and  $A \blacktriangleright -$ , for all  $A \in \mathbb{A}$ .

Theorems 3.1 and 3.2 below show that the Hilbert-type system for **LGL** is sound and complete with respect to the algebraic semantics. Proofs are given in [21] and further discussions of the use of algebraic structures for (commutative) bunched logic can be found in [50, 29].

**Theorem 3.1.** *The rules of  $\mathbf{LGL}_H$  are sound on layered algebras: for any layered algebra  $\mathbb{A}$ , for any interpretation  $\llbracket - \rrbracket : \text{Formulae} \rightarrow \mathbb{A}$ , and for any propositions  $\phi$  and  $\psi$ , if  $\phi \vdash \psi$  then  $\llbracket \phi \rrbracket \leq \llbracket \psi \rrbracket$ .*

**Theorem 3.2.** *For any propositions  $\phi$  and  $\psi$  of **LGL**, if  $\llbracket \phi \rrbracket \leq \llbracket \psi \rrbracket$  for all interpretations  $\llbracket - \rrbracket$  in all layered algebras, then  $\phi \vdash \psi$  in  $\mathbf{LGL}_H$ .*

**3.2. Bi-layering & Stacks.** The specific model of **LGL** for access control policy models that we develop in Section 4 requires worlds that are a particular class of layered graph called *bi-layered stacks*. Firstly, in order to identify this type of graph, we also require the notion of *bi-layering*.

Let  $\mathcal{G}$  be a graph and let  $\langle \mathcal{E}, \mathcal{F} \rangle$  be an ordered pair of finite sets of edges such that  $\mathcal{E}$  and  $\mathcal{F}$  are disjoint. The *bi-composition* operator,  $\hat{\textcircled{a}}_{\mathcal{E}, \mathcal{F}}$ , is a partial, binary operator on the set  $\text{Sg}(\mathcal{G})$ . We omit the subscripts on the operator where the distinguished sets of edges  $\mathcal{E}$  and  $\mathcal{F}$  are unambiguous. For all  $G_1, G_2 \in \text{Sg}(\mathcal{G})$ , the expression  $G_1 \hat{\textcircled{a}} G_2$  is defined just when both  $G_1 \textcircled{a}_{\mathcal{E}} G_2$  and  $G_2 \textcircled{a}_{\mathcal{F}} G_1$  are defined. When  $G_1 \hat{\textcircled{a}} G_2$  is defined, it is given by the vertex and edge sets

$$V(G_1 \hat{\textcircled{a}} G_2) = V(G_1) \cup V(G_2) \quad \text{and} \quad E(G_1 \hat{\textcircled{a}} G_2) = E(G_1) \cup E(G_2) \cup ((\mathcal{E} \cup \mathcal{F}) \cap \mathcal{G}[G_1, G_2]) .$$

We have a *bi-scaffold*  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle)$  if there is at least one pair of sub-graphs  $(G_1, G_2)$  such that  $G_1 \hat{\textcircled{a}} G_2$  is defined and if  $G_1 \textcircled{a}_{\mathcal{E}} G_2$  and  $G_2 \textcircled{a}_{\mathcal{F}} G_1$  are defined, then  $G_1 \hat{\textcircled{a}} G_2$  is *bi-layered* with respect to  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle)$ .

We can now consider the formation of bi-layered stacks via  $n$ -ary partial composition operations. Let  $\mathcal{G}$  be a graph and  $\mathcal{E}_1, \dots, \mathcal{E}_{n-1}, \mathcal{F}_1, \dots, \mathcal{F}_{n-1}$  be non-empty sets of edges of  $\mathcal{G}$ . Let the *strong bi-composition*  $\hat{\textcircled{a}}(G_1, \dots, G_n)$  of any subgraphs  $G_1, \dots, G_n$  of  $\mathcal{G}$  be defined just when  $V(G_i) \cap V(G_j) = \emptyset$ , for all  $i \neq j$ , and  $\mathcal{E}_i \subseteq \mathcal{G}(G_i, G_{i+1})$  and  $\mathcal{F}_i \subseteq \mathcal{G}(G_{i+1}, G_i)$ , for all  $i$ ; when it is defined, let

$$V(\hat{\textcircled{a}}(G_1, \dots, G_n)) = \bigcup_{1 \leq i \leq n} V(G_i) \quad \text{and}$$

$$E(\hat{\textcircled{a}}(G_1, \dots, G_n)) = \left( \bigcup_{1 \leq i \leq n} E(G_i) \right) \cup \left( \bigcup_{1 \leq i \leq n-1} \mathcal{E}_i \cup \mathcal{F}_i \right) .$$

The infix notation of  $G_1 \hat{\textcircled{a}}_{\mathcal{E}_1, \mathcal{F}_1} G_2 \hat{\textcircled{a}}_{\mathcal{E}_2, \mathcal{F}_2} \dots \hat{\textcircled{a}}_{\mathcal{E}_{n-2}, \mathcal{F}_{n-2}} G_{n-1} \hat{\textcircled{a}}_{\mathcal{E}_{n-1}, \mathcal{F}_{n-1}} G_n$  will also be used and in the case  $n = 1$  we define, as in [21],  $G = \hat{\textcircled{a}}(G)$ . Again, we will only write the subscripts on  $\hat{\textcircled{a}}$  where the edge sets need to be emphasized. Further details and examples are in [21] and naturally arising examples are in, for example, [46, 28, 48]. Illustrations of a bi-layered graph and a bi-layered stack are shown in Figures 5 and 6, respectively.

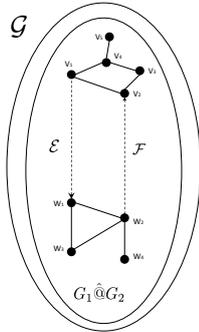
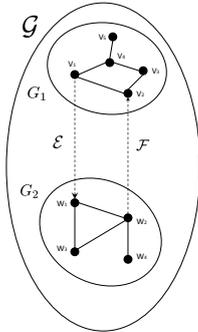


FIGURE 5. Bi-layered Graph

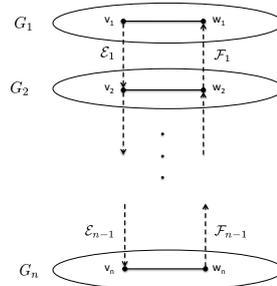


FIGURE 6. Bi-layered Stack

The  $n$ -ary composition defined above can be used to give a logical characterization of stacking. Details are found in [21] but the crucial change is the replacement of the multiplicative connectives with  $n$ -ary versions  $\blacktriangleright^n$  and  $\rightarrow^{n,m}$ , where  $n \geq 2$  and  $1 \leq m < n$ . In places, we use the notation  $\phi_1 \blacktriangleright \phi_2$  as an abbreviation for  $\blacktriangleright^2(\phi_1, \phi_2)$ . The satisfaction relations for the  $n$ -ary logical connectives are shown in Table 3 where, for simplicity, notation specifying scaffolds and the subscript on  $\models$  is omitted. As shown in Table 3, any world that satisfies the formula  $\rightarrow^{n,m}(\phi_1, \dots, \phi_{n-1}, \psi)$  must occur in the  $m$ th position of the  $n$ -ary composite satisfying  $\psi$ .

---

$G \models \blacktriangleright^n(\phi_1, \dots, \phi_n)$	iff	there exist $G_1, \dots, G_n$ such that $G = \hat{\@}(G_1, \dots, G_n)$ and $G_i \models \phi_i$ for $1 \leq i \leq n$
$G \models \rightarrow^{n,m}(\phi_1, \dots, \phi_{n-1}, \psi)$	iff	for all $G_1, \dots, G_{m-1}$ and $G_{m+1}, \dots, G_{n-1}$ $G_i \models \phi_i$ and $\hat{\@}(G_1, \dots, G_{m-1}, G, G_{m+1}, \dots, G_{n-1}) \downarrow$ implies $G_1 \hat{\@} \dots \hat{\@} G_{m-1} \hat{\@} G \hat{\@} G_{m+1} \hat{\@} \dots \hat{\@} G_{n-1} \models \psi$

---

TABLE 3. Satisfaction relation for  $n$ -ary connectives

**3.3. The Binary Case.** Almost all of the remainder of the paper will focus on the  $n$ -ary connective introduced above, since the theory to represent access control that we develop uses worlds that are bi-layered stacks. Discussion of the binary case (a 2-ary instance of  $n$ ) will be included where appropriate. The binary case is also used when presenting some examples but this has no effect on the formulation as it is shown in [21] that  $G_1 \hat{\@}_{\mathcal{E}, \mathcal{F}} G_2 \simeq G_1 \hat{\@}_{\mathcal{E}, \mathcal{F}} G_2$ , where  $\simeq$  denotes Kleene equality. In cases where  $n > 2$ , if  $G_1 \hat{\@}_{\mathcal{E}_1, \mathcal{F}_1} \dots \hat{\@}_{\mathcal{E}_{n-1}, \mathcal{F}_{n-1}} G_n \downarrow$  then  $G_1 \hat{\@}_{\mathcal{E}_1, \mathcal{F}_1} \dots \hat{\@}_{\mathcal{E}_{n-1}, \mathcal{F}_{n-1}} G_n \downarrow$  and both compositions are equal. The converse, however, does not hold in general due to the non-associativity of  $\hat{\@}$ .

Consider a graph,  $\mathcal{G}$ , consisting of four vertices, say  $v_1, v_2, v_3, v_4$  and precisely the eight edges indicated in Figure 7. Let  $G_i$  be the subgraph consisting of the single vertex  $\{v_i\}$ , for  $i = 1, \dots, 4$ .

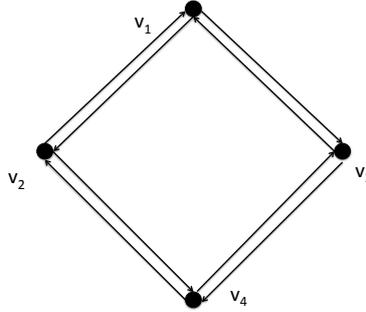


FIGURE 7. Bi-Layered Graph, but not a Stack

We can then consider the composite  $((G_1 \hat{\@} G_2) \hat{\@} G_3) \hat{\@} G_4$ , suppressing mention of the unique edge sets that make this defined. Many graphs can be construed as composites in this rather trivial way. For some examples, it is necessary to first adjoin edges so that each pair of vertices with precisely one edge between them has a pair of edges in opposite directions.

#### 4. AN ACCESS CONTROL POLICY MODEL OF LGL

Access control policy models — such as Bell-LaPadula, Biba, and Chinese Walls, as discussed in Section 1 — are concerned with regulating how agents access and manipulate data items. These models are usually presented in terms of the security designations of data items (e.g., ‘no read-up, no write-down’, and so on). However, the agents and data items concerned are located

within system architectures and the associated security designations are implemented within those systems. As mentioned in the introduction, in our approach, we distinguish between the security architecture, as specified by the security policy, and the underlying system architecture, to which the security policy is intended to apply. This approach allows, for example, one to consider whether the security architecture and the underlying system architecture are compatible with one another. For example, it may be that there are features of the system architecture that undermine the security architecture. Example 4.5 illustrates this point.

Generally, we would emphasise that security designation is a synthetic attribute and is just one characteristic attribute for exercising security control (e.g., in authentication, prior to access control). Other attributes may also be used, and location, which may be logical or physical, is one that is commonly used. Indeed, the implementation of a control may depend on location; for example, a mobile device on a corporate network may have direct access to confidential data, but may require a VPN connection when connected to the internet via an external network, even if in the same physical location. Moreover, vulnerabilities may arise if it is assumed that all agents at a given location will have a particular designation: if a passenger gains unauthorized access to an airline's lounge via a back door lacking appropriate access control, then she may obtain services to which she is not entitled according to the policy. That is, there may be a mismatch between policy (formulated explicitly using designation) and implementation (done without explicit designation checks). We are interested not merely in the fact that the system may not be secure (according to policy), but in exactly what way.

In this section, we establish a theory of **LGL** that describes how access control policies regulate how located agents access and manipulate located data items. To this end, we establish a class of atomic formulae that denote an agent's access to a data item along a path within the system's architecture.

The languages and logics introduced below are, at present, only given a semantics on stacks. These are linear structures. Thus the languages are not currently able to account for all lattice models of security levels.

Let  $G = G_1 \hat{\otimes}_{\mathcal{E}_1, \mathcal{F}_1} G_2 \hat{\otimes}_{\mathcal{E}_2, \mathcal{F}_2} \dots \hat{\otimes}_{\mathcal{E}_{n-2}, \mathcal{F}_{n-2}} G_{n-1} \hat{\otimes}_{\mathcal{E}_{n-1}, \mathcal{F}_{n-1}} G_n$  be a bi-layered stack and let the index of each layer identify a security classification with 1 the highest down to the lowest value  $n$ . We restrict ourselves to stacks that do not contain additional, extraneous, distinguished edges. Graphs of this type may be present in the ambient structure but are not considered. Let  $\mathcal{A}$  be a set of *agents* and  $\mathcal{D}$  be a set of *data items*. The sets  $\mathcal{A}$  and  $\mathcal{D}$  are disjoint. We use  $A$  and  $D$  to denote agents and data items, respectively. Let  $\mathbf{Atom}$  be a set of atomic actions. For example, atomic actions might include, among others, **read** and **write**, or more system specific actions, such as moving data items between locations. Let the set of actions be  $\mathbf{Act} = \mathcal{P}(\mathcal{A}) \times \mathbf{Atom} \times \mathcal{D}$ . A triple of the form  $(\{A\}, \mathbf{write}, D)$  will be denoted as  $A \mathbf{write} D$ , which is read as 'agent  $A$  writes to data item  $D$ '.

Agents and data items are to be used to define a new family of atoms, with notation  $\mapsto$  (suitably indexed), that we call *access operators* or *access assertions*. These have some similarity to the *points-to* relation of Pointer and Separation Logic, [37, 52]. Access is represented in terms of an agent and piece of data meaning that an instance of a relation  $A \mapsto D$  is to be read as 'agent  $A$  has access to data item  $D$ '. The operator is given a superscript relating to actions and we use **r** and **w** throughout to denote read and write actions. An instance of the relation  $A \mapsto^{\mathbf{r}} D$  means that agent  $A$  has read access to  $D$ . Where the superscript of  $\mapsto$  is used it is intended to denote either read or write (or both). Notational similarities between the access operator and the points-to relation are not indicative of a technical connection but arise as a result of using points-to as inspiration.

Note that the access operators represent, in a sense, the privilege to perform actions at a single moment of time, rather than the (state change arising from) the action itself. In Section 6 we will add the ability to reason about the state change arising from the action. This allows for a distinction between actions that, viewed statically, could be performed because of appropriate privilege, and actions that may be performed in a given system taking into account dynamic changes of state. In much of the existing literature on formal security policy models (e.g., [5]) the privilege to perform actions in a given state is represented, as is the the action itself in the form of

a state-transition, allowing for a ‘Basic Security Theorem’ to be proved showing the preservation of secure state under policy.

We will use the notion of a *path* within a stack  $G$ , [9, 25]. A path  $P$  is a (sub)graph where  $V(P)$  is a sequence of vertices  $v_1, \dots, v_n$  and  $E(P) = \{(v_i, v_{i+1}) \mid 1 \leq i \leq n-1\}$ . That is,  $E(P)$  only contains edges that connect subsequent vertices in the sequence  $V(P)$ . We will denote paths by a sequence of vertices  $P = v_1, v_2, \dots, v_n$  and permit an empty path where  $v_i = v_j$  for all  $i \neq j$  and  $E(P) = \emptyset$ . Multi-graphs are not permitted. That is, there is at most one edge from one vertex to another.

Edges of  $G$  will be labelled by *Access Control Lists* (ACLs): an ACL will here consist of a finite set of triples of the form  $(A, \mathbf{a}, D) \in \mathcal{A} \times \mathbf{Atom} \times \mathcal{D}$ . Each edge  $e$  is labelled by a unique access control list  $\text{Label}(e)$ . If  $(A, \mathbf{a}, D) \in \text{Label}(e)$ , this means that  $A$  can perform an (atomic) action  $\mathbf{a}$  on data item  $D$ , via edge  $e$ .

In what follows, we define a satisfaction relation for logical formulae at worlds which consist of bi-layered stacks and mappings of agents and data to vertices of the stack. Formally, a stack of the form  $G = G_1 \hat{\otimes}_{\mathcal{E}_1, \mathcal{F}_1} \dots \hat{\otimes}_{\mathcal{E}_{n-1}, \mathcal{F}_{n-1}} G_n$  consists of the sequence  $G_1, \dots, G_n$  of components (and the ambient graph and distinguished edge sets). Thus, a formal notation should really keep track of this sequence of components. However, in what follows, we develop a shorthand in which we simply write instances of satisfaction in the form  $F, G \models \phi$ , where  $G$  is the composite stack (for example, as immediately above),  $F$  is an assignment that describes which agents and data lie at which vertices of  $G$ , and  $\phi$  is a formula; note that this mentions only the composite  $G$  and not its components. In reading this notation, it is important to remember that the graph component is an abuse that stands in for the whole structure of the stack. The interpretation that we give demands that worlds carry certain (non-commutative) composition operations. In particular, this requires composition operators on stacks. Proposition 4.2 shows that these composition operations on stacks are induced from those on graphs in a very direct fashion. This, together with a notion of composition for pairs of agent and mapping assignments, justifies our abuse of notation, but we make select further comment on the notation below for the sake of clarity.

The language **LGL** was previously interpreted on the set of subgraphs of an ambient graph,  $\mathcal{G}$ . Below, it will be interpreted using stacks. However, the new atoms given by access operators require, in addition, the specification of where data and agents lie.

**Definition 4.1.** Let  $\mathcal{G}$  be a given ambient graph. Define an *agent-data assignment* to be a partial function from the (disjoint) union of the sets of agents and data to the set of vertices of  $\mathcal{G}$ . We reserve the letter  $F$  for such assignments. The *domain* of  $F$  is the (largest) set of elements on which  $F$  is defined. Let  $\mathbb{F}$  be the set of all agent-data assignments (for the given  $\mathcal{G}$ ).

Abusing notation slightly, we define an  $n$ -ary composition  $\hat{\otimes}$  for elements  $F_1, \dots, F_n$  of  $\mathbb{F}$ . Let  $\hat{\otimes}(F_1, \dots, F_n)$  be defined if and only if the domains of  $F_1, \dots, F_n$  are disjoint; when it is defined, let the graph of this partial function be the union of the graphs of  $F_1, \dots, F_n$ .

We will also use the infix notation  $F_1 \hat{\otimes} \dots \hat{\otimes} F_n$ . An illustration for the binary case, denoted  $\hat{\otimes}$  in line with the notation of Section 3, is shown in Figure 8.

Now consider pairs of the form  $(F, G)$ , where  $G$  is a stack in  $\mathcal{G}$  and  $F$  is an agent-data assignment. Define an agent-data assignment  $F$  to be *compatible* with a sub-graph  $G$  of the ambient graph  $\mathcal{G}$  when the image of  $F$  is contained in the set of vertices of  $G$ . The  $n$ -ary composite  $\hat{\otimes}((F_1, G_1), \dots, (F_n, G_n))$  of pairs  $(F_1, G_1), \dots, (F_n, G_n)$  is then defined to be the pair  $(\hat{\otimes}(F_1, \dots, F_n), \hat{\otimes}(G_1, \dots, G_n))$ , where both of the components of this pair are defined; the composite is undefined if either of the components of this pair are undefined. This family of composites underpins the interpretation of multiplicative connectives given below.

Note that if a graph  $G$  is a stack, it can be decomposed into its layers, say  $G_1, \dots, G_n$ . Any agent-data assignment  $F$  that is compatible with  $G$  can then be decomposed into  $F_1, \dots, F_n$  such that each  $F_i$  is compatible with the corresponding  $G_i$ : we simply take each  $F_i$  to be the restriction of  $F$  to the set of agents and data that are mapped by  $F$  into  $G_i$ .

Let  $G$  be a stack with layers  $G_1, \dots, G_n$ . Let  $F$  be an agent-data assignment that is compatible with  $G$ . Define the corresponding *agent assignment* to be the restriction of  $F$  to  $\mathcal{A}$ . This will be written as  $\alpha^F$  or simply  $\alpha : \mathcal{A} \rightarrow V(G)$  when  $F$  is clear from the context. Each vertex of  $G$  is in a unique layer  $G_i$  of  $G$ . For  $\alpha(A) \in V(G_i)$ , define  $i$  to give the *clearance level* of agent  $A$ .

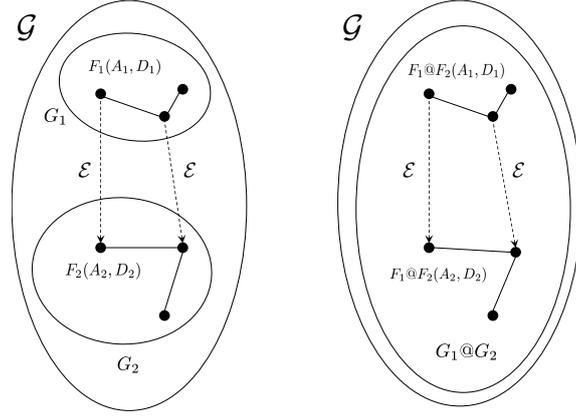


FIGURE 8. Illustration of Definition 4.1 (binary case)

Define the *data assignment* corresponding to  $F$  to be the restriction of  $F$  to  $\mathcal{D}$ . This will be written as  $\delta^F$ , or simply  $\delta : \mathcal{D} \rightarrow V(G)$  be a partial function from data items  $D$  to a vertex of  $G$ . Define  $i$  to be the *clearance level* of data item  $D$ , if  $\delta(D) \in V(G_i)$ .

Note that  $F$  is given by a unique pair  $(\alpha, \delta)$ . Let  $\alpha[\mathcal{A}]$  (that is,  $F[\mathcal{A}]$ ) and  $\delta[\mathcal{D}]$  (that is,  $F[\mathcal{D}]$ ) denote the images of  $\mathcal{A}$  and  $\mathcal{D}$  under  $\alpha$  and  $\delta$ , respectively.

It is convenient at this point to begin to discuss some special kinds of atomic formulae that are available, given the structure of worlds. The atomic formulae  $q$  now include not only basic propositional letters  $p$ , as in **LGL**, but also the access assertions (also referred to as access operators):

$$q ::= p \mid A \mapsto^a D \mid A \mapsto_{\uparrow}^a D \mid A \mapsto_{\downarrow}^a D.$$

The assertion  $A \mapsto^a D$  says that the agent  $A$  is able to access the data  $D$ . Accessing data may involve reading, writing, or otherwise manipulating (perhaps sharing) the data.

Recall that the worlds satisfying access clauses are bi-layered stacks,  $G = G_1 \hat{\otimes}_{\mathcal{E}_1, \mathcal{F}_1} \dots \hat{\otimes}_{\mathcal{E}_{n-1}, \mathcal{F}_{n-1}} G_n$ . The semantics of the  $A \mapsto^a D$  assertion is

$$(1) \quad F, G \models A \mapsto^a D \quad \text{iff} \quad \text{there is a non-empty path } P = F(A), \dots, F(D) \text{ in } G \\ \text{such that } (A, a, D) \in \text{Label}(e) \text{ for all } e \in E(P).$$

The access operator subscripts  $\uparrow$  and  $\downarrow$  to indicate whether the action is taking place upwards or downwards. The assertion  $A \mapsto_{\uparrow}^a D$  says that the action  $a$  is (*strictly*) *upward*. The semantics of this assertion is given by

$$(2) \quad F, G \models A \mapsto_{\uparrow}^a D \quad \text{iff} \quad \text{there is a non-empty path } P = F(A), \dots, F(D) \text{ in } G \\ \text{such that } E(P) \subseteq \bigcup_k \mathcal{F}_{1 \leq k \leq n-1} \text{ and,} \\ (A, a, D) \in \text{Label}(e) \text{ for all } e \in E(P).$$

So, the added condition on edges of the path ensures that the path is strictly monotonic, moving upwards only. Again, all edges of  $P$  must be labelled with the appropriate ACLs.

The assertion  $A \mapsto_{\downarrow}^a D$  says that the action  $a$  (*strictly*) *downward* action. The semantics of this assertion is

$$(3) \quad F, G \models A \mapsto_{\downarrow}^a D \quad \text{iff} \quad \text{there is a non-empty path } P = F(A), \dots, F(D) \text{ in } G \\ \text{such that } E(P) \subseteq \bigcup_k \mathcal{E}_{1 \leq k \leq n-1} \text{ and,} \\ (A, a, D) \in \text{Label}(e) \text{ for all } e \in E(P).$$

Let the ambient graph be  $\mathcal{G}$  and the distinguished edge sets be  $\mathcal{E}_1, \dots, \mathcal{E}_n$  and  $\mathcal{F}_1, \dots, \mathcal{F}_n$ . A ‘stack’ will now be taken to be a strong bi-composition of the form

$$G_1 \hat{\otimes}_{\mathcal{E}_{i(1)}, \mathcal{F}_{i(1)}} \dots \hat{\otimes}_{\mathcal{E}_{i(m)}, \mathcal{F}_{i(m)}} G_m,$$

where  $i(j) \neq i(k)$ , for all  $j \neq k$ , and  $1 \leq i(j) \leq n$ , for all  $j$ . Thus, this composite simply uses a sub-family of the original family of (pairs of) distinguished edge sets. Let  $\text{stack}(\mathcal{G})$  be the set of such stacks. Let  $\text{compatible}(\mathbb{F}, \mathcal{G})$  be the set of all pairs  $(F, G) \in \mathbb{F} \times \text{Sg}(\mathcal{G})$ , where  $G$  is a stack in

$F, G \models \top$ always	$F, G \models \perp$ never	$F, G \models p$ iff $F, G \in \mathcal{V}(p)$
$F, G \models \phi \wedge \psi$	iff	$F, G \models \phi$ and $F, G \models \psi$
$F, G \models \phi \vee \psi$	iff	$F, G \models \phi$ or $F, G \models \psi$
$F, G \models \phi \rightarrow \psi$	iff	$F, G \models \phi$ implies $F, G \models \psi$
$F, G \models \blacktriangleright^n(\phi_1, \dots, \phi_n)$	iff	there exist $F_1, G_1, \dots, F_n, G_n$ such that $G = \hat{\@}(G_1, \dots, G_n), F = \hat{\@}(F_1, \dots, F_n)$ and $F_i, G_i \models \phi_i$ for $1 \leq i \leq n$
$F, G \models \blacktriangleright^{n,m}(\phi_1, \dots, \phi_{n-1}, \psi)$	iff	for all $F_1, G_1, \dots, F_{m-1}, G_{m-1}$ and $F_{m+1}, G_{m+1}, \dots, F_{n-1}, G_{n-1},$ $F_i, G_i \models \phi_i$ and $\hat{\@}(F_1, \dots, F_{m-1}, F, F_{m+1}, \dots, F_{n-1}) \downarrow$ and $\hat{\@}(G_1, \dots, G_{m-1}, G, G_{m+1}, \dots, G_{n-1}) \downarrow$ implies $\hat{\@}(F_1, \dots, F, \dots, F_{n-1}), \hat{\@}(G_1, \dots, G, \dots, G_{n-1}) \models \psi$

 TABLE 4. Satisfaction Relation for Access Control (Basic **LGL**)

$F, G \models \phi_1 \blacktriangleright \phi_2$	iff	there exist $F_1, G_1, F_2, G_2$ such that $G = G_1 \hat{\@} G_2, F = F_1 \hat{\@} F_2$ and $F_1, G_1 \models \phi_1$ and $F_2, G_2 \models \phi_2$
$F, G \models \phi \blacktriangleright \psi$	iff	for all $J, H$ such that $G \hat{\@} H \downarrow$ and $F \hat{\@} J \downarrow,$ $J, H \models \phi$ implies $F \hat{\@} J, G \hat{\@} H \models \psi$
$F, G \models \phi \blacktriangleleft \psi$	iff	for all $J, H$ such that $H \hat{\@} G \downarrow$ and $J \hat{\@} F \downarrow,$ $J, H \models \phi$ implies $J \hat{\@} F, H \hat{\@} G \models \psi$

TABLE 5. Satisfaction Relation for Binary Versions of Multiplicative Connectors

$\mathcal{G}$ , and where  $F$  is compatible with  $G$ . We suppress here the mention of the distinguished edge set pairs  $((\mathcal{E}_1, \mathcal{F}_1), \dots, (\mathcal{E}_n, \mathcal{F}_n))$  in the stack and compatible notation.

For  $\mathbb{F}$  as above, a valuation function

$$\mathcal{V} : \text{Atoms} \rightarrow \mathcal{P}(\text{compatible}(\mathbb{F}, \mathcal{G}))$$

generates a semantics for the model. The satisfaction relation is now

$$\models \subseteq \text{compatible}(\mathbb{F}, \mathcal{G}) \times \text{Formulae}$$

as defined by Tables 4 and 6. Binary instances of the multiplicative connectives are presented in Table 5 to further explain the more general  $n$ -ary multiplicatives, and because binary cases often arise in examples.

For any proposition  $\phi$ , let  $\llbracket \phi \rrbracket = \{F, G \mid F, G \models \phi\}$ . This defines an interpretation function

$$\llbracket - \rrbracket : \text{Formulae} \rightarrow \mathcal{P}(\text{compatible}(\mathbb{F}, \mathcal{G})).$$

The monoidal structure of  $\text{compatible}(\mathbb{F}, \mathcal{G})$  discussed above ensures that  $\mathcal{P}(\text{compatible}(\mathbb{F}, \mathcal{G}))$  is a layered algebra.

Note that the  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle)$  notation of **LGL** for bi-layering and stacks [21], including the subscript on  $\models$ , is suppressed for simplicity (this additional data is implicit), and that the access operators are the only formulae that require  $F$  for interpretation.

The clauses in Table 4 present the satisfaction for the  $n$ -ary connectives with satisfaction for the binary case shown in Table 5. Given that the worlds that satisfy formulae in this specific

model are stacks, satisfaction requires an implicit enumeration of the decomposition of a stack  $G$  (e.g.,  $G = G_1 \hat{\otimes} \dots \hat{\otimes} G_n$ ). As noted above, the specific decomposition of the stack occurring in an instance of this relation is suppressed.

In the case of the multiplicative connector  $\blacktriangleright$ ,  $G$  must decompose into stacks (not merely subgraphs) that satisfy the respective parts of the formula. They will (implicitly) have their own specific decompositions into layers. So if, for example,  $F, G \models \phi_1 \blacktriangleright \phi_2$  then, assuming appropriate functions, for  $F_1, G_1 \models \phi_1$  and  $F_2, G_2 \models \phi_2$ ,  $G_1$  and  $G_2$  must also be stacks. The stacks  $G_1$  and  $G_2$  are constructed based on the existence of a specific, but non-unique, decomposition of  $G$ . That is,  $G_1$  and  $G_2$  are the composition of a subsequence of the  $n$  layers of  $G$  that allows properties  $\phi_1$  and  $\phi_2$  to be satisfied. A decomposition of  $F$ , yielding  $F_1$  and  $F_2$  that are compatible with  $G_1$  and  $G_2$ , respectively, can be found by restricting  $F$  to  $G_1$  and  $G_2$ , respectively.

Proposition 4.2, which is a minor adaptation of a result found in [21], shows that stacks have a well-defined composition operation. As noted above, this extends to a well-defined composition operation on compatible pairs, and this supports the satisfaction clauses for  $\blacktriangleright^n$  and  $\rightarrow^{n,m}$ .

**Proposition 4.2.** *Let  $\mathcal{G}$  be a graph, let all  $G_i$  below be non-empty subgraphs of  $\mathcal{G}$ , and all  $\mathcal{E}_i, \mathcal{F}_i$  be non-empty sets of edges of  $\mathcal{G}$ . Let  $G_1 \hat{\otimes}_{\mathcal{E}_1, \mathcal{F}_1} \dots \hat{\otimes}_{\mathcal{E}_{n-1}, \mathcal{F}_{n-1}} G_n$  be defined. Then, for  $1 \leq k \leq m \leq n$ ,  $G_k \hat{\otimes}_{\mathcal{E}_k, \mathcal{F}_k} \dots \hat{\otimes}_{\mathcal{E}_{m-1}, \mathcal{F}_{m-1}} G_m$  is defined. That is, if an  $n$ -ary strong composite is defined with respect to a sequence of  $n-1$  edge sets, then the evident  $(1+m-k)$ -ary strong composite is defined for any consecutive subsequence. In particular, this holds for  $m = k+1$ , so that  $G_k \hat{\otimes}_{\mathcal{E}_k, \mathcal{F}_k} G_{k+1}$  is defined. Moreover,*

(4)

$$G_1 \hat{\otimes}_{\mathcal{E}_1, \mathcal{F}_1} \dots \hat{\otimes}_{\mathcal{E}_{n-1}, \mathcal{F}_{n-1}} G_n = (G_{k_0} \hat{\otimes}_{\mathcal{E}_{k_0}, \mathcal{F}_{k_0}} \dots \hat{\otimes}_{\mathcal{E}_{k_1-1}, \mathcal{F}_{k_1-1}} G_{k_1}) \hat{\otimes}_{\mathcal{E}_{k_1}, \mathcal{F}_{k_1}} \dots \hat{\otimes}_{\mathcal{E}_{k_z-1}, \mathcal{F}_{k_z-1}} (G_{k_z} \hat{\otimes}_{\mathcal{E}_{k_z}, \mathcal{F}_{k_z}} \dots \hat{\otimes}_{\mathcal{E}_{k_{z+1}-1}, \mathcal{F}_{k_{z+1}-1}} G_{k_{z+1}}),$$

for all  $1 = k_0 \leq k_1 \leq \dots \leq k_z \leq k_{z+1} = n$  and  $0 \leq z \leq n$  such that the right-hand-side ‘partitions’ the sequence from the left-hand-side into consecutive subsequences (each expression in brackets). Each of the bracketed expressions  $G_{k_i} \hat{\otimes}_{\mathcal{E}_{k_i}, \mathcal{F}_{k_i}} \dots \hat{\otimes}_{\mathcal{E}_{k_{i+1}-1}, \mathcal{F}_{k_{i+1}-1}} G_{k_{i+1}}$  on the right-hand-side is intended to be a  $(1+k_{i+1}-k_i)$ -ary composite.

The proof is given in [21] for the case involving edge sets  $\mathcal{E}_1, \dots, \mathcal{E}_{n-1}$ . The inclusion of the additional edge sets  $\mathcal{F}_i$  means only a trivial change.

**Example 4.3.** Let  $G = G_1 \hat{\otimes}_{\mathcal{E}_1, \mathcal{F}_1} G_2 \hat{\otimes}_{\mathcal{E}_2, \mathcal{F}_2} G_3 \hat{\otimes}_{\mathcal{E}_3, \mathcal{F}_3} G_4$ . That is,  $G$  is a bi-layered stack with 4 layers. Let  $H_1 = G_1 \hat{\otimes}_{\mathcal{E}_1, \mathcal{F}_1} G_2$  and let  $H_2 = G_3 \hat{\otimes}_{\mathcal{E}_3, \mathcal{F}_3} G_4$ . Clearly  $H_1$  and  $H_2$  are defined and  $G = H_1 \hat{\otimes}_{\mathcal{E}_2, \mathcal{F}_2} H_2$ , showing that  $G$  can be decomposed into stacks of fewer layers.  $\square$

---

$F, G \models A \mapsto^{\mathbf{a}} D$	iff	there is a non-empty path $P = F(A), \dots, F(D)$ in $G$ such that $(A, \mathbf{a}, D) \in \text{Label}(e)$ for all $e \in E(P)$ .
$F, G \models A \mapsto_{\mathbf{0}}^{\mathbf{a}} D$	iff	there is a non-empty path $P = F(A), \dots, F(D)$ in $G$ such that $E(P) \subseteq \bigcup_k \mathcal{F}_{1 \leq k \leq n-1}$ and, $(A, \mathbf{a}, D) \in \text{Label}(e)$ for all $e \in E(P)$ .
$F, G \models A \mapsto_{\mathbf{0}}^{\mathbf{a}} D$	iff	there is a non-empty path $P = F(A), \dots, F(D)$ in $G$ such that $E(P) \subseteq \bigcup_k \mathcal{E}_{1 \leq k \leq n-1}$ and, $(A, \mathbf{a}, D) \in \text{Label}(e)$ for all $e \in E(P)$ .

---

TABLE 6. Satisfaction Relation for Access Control Assertions

The non-uniqueness of the decomposition of a stack satisfying a formula involving  $\blacktriangleright$  is illustrated by the following example:

**Example 4.4.** Consider Figure 9. Let  $G = G_1 \hat{\otimes} \dots \hat{\otimes} G_5$  and assume the existence of appropriate functions  $F_i$  that map the agents and data items into  $G$  as shown. Also assuming the existence

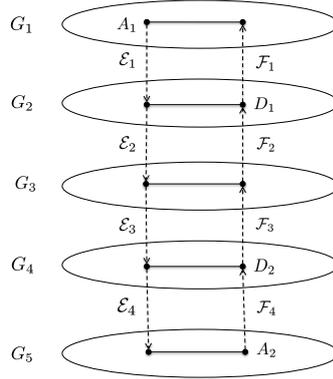


FIGURE 9. Illustration of Example 4.4

of appropriate ACLs, it is clear from the diagram that  $F, G \models (A_1 \mapsto_D^a D_1) \blacktriangleright (A_2 \mapsto_U^a D_2)$ , where  $F = \hat{\text{@}}(F_1, \dots, F_5)$ . Now let  $H_1 = G_1 \hat{\text{@}}_{\mathcal{E}_1, \mathcal{F}_1} G_2$ , let  $H_2 = G_3 \hat{\text{@}}_{\mathcal{E}_3, \mathcal{F}_3} G_4 \hat{\text{@}}_{\mathcal{E}_4, \mathcal{F}_4} G_5$ , let  $H'_1 = G_1 \hat{\text{@}}_{\mathcal{E}_1, \mathcal{F}_1} G_2 \hat{\text{@}}_{\mathcal{E}_2, \mathcal{F}_2} G_3$  and let  $H'_2 = G_4 \hat{\text{@}}_{\mathcal{E}_4, \mathcal{F}_4} G_5$  with  $J_1, J_2, J'_1, J'_2$  denoting the corresponding composition of functions. Then it is the case that both of the following hold:

$$\begin{aligned} J_1 \hat{\text{@}}_{\mathcal{E}_2, \mathcal{F}_2} J_2, H_1 \hat{\text{@}}_{\mathcal{E}_2, \mathcal{F}_2} H_2 &\models (A_1 \mapsto_D^a D_1) \blacktriangleright (A_2 \mapsto_U^a D_2) \text{ and} \\ J'_1 \hat{\text{@}}_{\mathcal{E}_3, \mathcal{F}_3} J'_2, H'_1 \hat{\text{@}}_{\mathcal{E}_3, \mathcal{F}_3} H'_2 &\models (A_1 \mapsto_D^a D_1) \blacktriangleright (A_2 \mapsto_U^a D_2). \end{aligned}$$

□

**4.1. Examples of Access Control Policy Models.** It is tempting ask whether our **LGL**-based approach to modelling systems and their associated access control policy models — that is, access control policy models of **LGL** — is able to present all instances of Bell La Padula, Biba, and Chinese Wall (or Brewer Nash) policies. There are essentially two questions here. First, can the **LGL**-based approach express all instances of these policies expressed at the (finite) level of abstraction taken in, say, [3] or [4]? Second, given our motivations as expressed in Sections 1 and 2, can we identify classes of distributed systems and associated access control policies that are uniformly and completely expressible using our **LGL**-based approach?

The answer to the first question is, up to some straightforward choices about the design of models, quite trivially yes. At the given level of abstraction, it is immediate that **LGL** with the access operator(s) as described in Section 4 can capture the various read and write constraints required for any instance of the Bell La Padula and Biba models. Lattice models in general require the binary case, as mentioned in Section 3.3.

Similarly, it is immediate that the multiplicative conjunction, as introduced in Section 5, next, is sufficient to describe the separated regions of a system that Chinese Walls create. However, given our motivations, as explained in Sections 1 and 2, this question is not of much further interest.

To address the second question, which is very far from trivial, it would seem to be necessary to develop a more substantially detailed view of class of underlying distributed systems to which the policy models would be intended to apply, formulate an **LGL**-based model of that class of systems, and then formulate a general definition of each class of policy model in that context. This would be a very substantial piece of work, and is certainly beyond the scope of this paper. We can, however, briefly discuss some of the issues that would need to be addressed in such a programme of work.

- First, any such class of distributed systems and associated models would necessarily be somewhat restrictive. Identifying appropriate choices — sufficiently general for an interesting result, yet sufficiently tractable — would already be a significant challenge.
- Second, even given a class of systems of interest, together with a viable class of models, a number of choices concerning the relationship between security features, as described by access control policies, and operational features, which may undermine the intended access

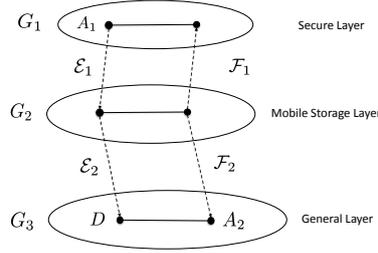


FIGURE 10. Organizational Structure including Mobile Storage Layer

control policies, would typically remain. Consequently, further modelling work — in the established sense of observation, model construction, model exploration, and validation — would be required in order to identify appropriate classes of systems and policies to be studied.

- Third, it is unlikely that the simple propositional form of **LGL** presented here will prove sufficiently expressive to be a convenient or even adequate tool for the purpose. It is likely that (at least) an extension with ‘resource modalities’, along the lines of the extension discussed briefly in [21] will be required. In the weak substructural setting of **LGL**, such an extension will require a substantial meta-theoretical development.

Such a general analysis of the expressivity of **LGL**-based approaches would be intended to help identify a modelling methodology

Having outlined a rather substantial programme of research about expressivity and methodology — the execution of which would be well beyond this first paper — we now present a range of relatively simple examples of how the somewhat elementary version of the approach developed so far can be applied. A recurrent feature of the examples is compositionality. For example, composition of policies as in Example 4.7, which occurs at the level logical propositions, or the compositional structure of system architecture as in Examples 4.5, 4.6, 4.9, or 4.8, which occurs at the level of the logic’s models. As we have remarked in Section 2, Sections 5 and 6 describe yet further forms of compositionality.

**Example 4.5** (Side-channel). This example, which was introduced in Section 1, gives a first indication of why it is important that access control policies take account of organizational structure.

Consider an organization whose network contains two clearance levels: secure and the lower level general. It is the intention of the organization that data is not permitted to be written from the secure level to the general level, as this may compromise the confidentiality of the data. The organization chooses to enforce this by implementing a no write-down rule, according to clearance levels. This policy, based on the traditional method of using clearance levels only, is inadequate and could lead to a violation in policy if the underlying organizational structure has not been considered. The widespread usage of mobile storage devices adds an additional layer, the Mobile Storage Layer, to the organizational structure meaning that data could be written from the secure to the general layer.

Consider Figure 10, which is similar to Figure 1, showing the structure of the organization. Let  $G_1, G_2, G_3$  represent the Secure, Mobile Storage, and General layers. Suppose that  $\mathcal{G} = G = G_1 \hat{\otimes}_{\mathcal{E}_1, \mathcal{F}_1} G_2 \hat{\otimes}_{\mathcal{E}_2, \mathcal{F}_2} G_3$ , forming a stack. Let  $F$  be the function that maps agents  $A_1, A_2$  and data item  $D$  into the respective layers as shown. Suppose that there are no restrictions placed on data being written to, or from, mobile devices. Then the existence of appropriate ACLs around  $G$  may then be assumed.

The current policy, as specified by the organization, means that as there are no restrictions based on this structure. Therefore, agent  $A_1$  can write along the edge of  $\mathcal{E}_1$  into  $G_2$  (the Mobile Storage Layer). A second write action could then be executed along  $\mathcal{E}_2$ , meaning that secure data has been written from  $G_1$  into  $G_3$ . This sequence of potential write actions is represented the following judgement:

$$F, G \vDash A_1 \mapsto_D^{\mathfrak{a}} D.$$

This shows that  $A_1$ , in the secure layer, does have write-down access to data in the general layer. As a result, the policy of the organization has been violated.

The organization can correct the flaw in its policy by preventing data from being written to a mobile storage device. In practice this could be achieved by, for example, blocking the USB ports of agents with a secure clearance level. If the organization does take account of the additional structure and enforce new access conditions, the original goal that no data should be written from the Secure layer ( $G_1$ ) to the General layer ( $G_3$ ), will be attained. Assuming this happens, the policy can be represented by

$$F, G \models \neg(A_1 \mapsto_D^w D).$$

So we can see that the organization in question, which failed to take account of its underlying structure, was unable to write a policy to adequately enforce its wishes. Taking account of the existence of the Mobile Storage Layer means that a more appropriate rule can be implemented.  $\square$

**Example 4.6** (Data Access within a Hospital). This very simple example is about access to data within a hospital. In this somewhat simplified setting, we assume three types of agents and data: medical, administrative, and technical. In this simple example, we do not address issues such as privacy or access to anonymized data: whilst of interested, such concerns are beyond our present scope.

Suppose that medical staff and data are assigned a clearance level of 1 while all other staff and data are assigned clearance level 2. Administrative data might be, for example, patient appointments while technical data may concern maintenance schedules for equipment. Medical staff should not be involved in the setting of appointments or in decisions about scheduling of maintenance and as a result should be restricted by a no write-down rule. In addition, technicians have no need to access patients' medical records so should be under a no read-up restriction. For this reason, the BLP model is implemented across the hospital. This 'blanket' implementation is consistent with the approach in previous presentations (e.g., a lattice model) of access control, where access is governed solely by clearance level.

Consider now the administrative staff. It is standard practice in many medical establishments that administrative staff perform tasks relating to medical records, such as issuing test results. For this reason, an implementation of BLP based purely on clearance level will inhibit the ability of these agents to perform their duties. Recognising this, a hospital may wish to write a new policy so that administrative staff can be given access to specific pieces of medical data. This, however, cannot be achieved while access restrictions are based solely on a clearance level. In order to design an appropriate policy, each member of staff will require specific, and possibly individual, access privileges. For example, consultants in a hospital often have dedicated secretaries who should only have access to medical data concerning patients of the given consultant. A similar situation could be described where technicians work solely on a particular piece of equipment. Access conditions specified in this manner will be very difficult to manage and potentially counter-productive. Agents may have to be assigned a clearance level that is unique so that the relevant access permissions can be specified. That is, clearance levels are singleton sets associated with one agent. In such a situation it would be impossible to, for example, provide holiday cover as access rights are different for each individual agent. A more effective solution is to design a policy based on the structure of the system model, allowing specific access rights to be defined.

Let  $G = G_1 \hat{\otimes} G_2$  be a bi-layered stack representing the system model of a hospital. Let the individual layers correspond to clearance levels 1 and 2. Let  $A_M$  denote medical staff, let  $A_A$  denote administrative staff and let  $A_T$  denote technicians. Similarly, let  $D_M, D_A$  and  $D_T$  denote medical, administrative and technical data, respectively. Let  $F$  be the function that maps all agents and data items into the layered graph.

Consider Figure 11. An effective access policy would mean that agent  $A_A$  has access to  $D_M$  while  $A_T$  does not. Given that there are paths between the locations of both  $A_A$  and  $A_T$  to  $D_M$ , satisfaction of the access operator relies on appropriate ACL labels. Let  $P_1$  and  $P_2$  denote the (by assumption, unique) paths to  $F(D_M)$  from  $F(A_A)$  and  $F(A_T)$ , respectively. If the edges of  $G$  are labelled such that  $(A_A, \mathbf{r}, D_M) \in \text{Label}(e)$  for all  $e \in E(P_1)$  but  $(A_T, \mathbf{r}, D_M) \notin \text{Label}(e)$  for some  $e \in E(P_2)$ , then the access rights of the system can be given as follows:

$$F, G \models A_A \mapsto_U^x D_M \quad \text{and} \quad F, G \models \neg(A_T \mapsto_U^x D_M).$$

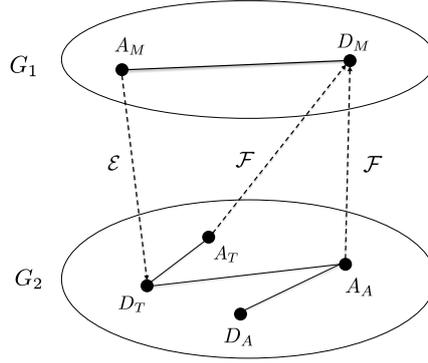


FIGURE 11. Location of Agents and Data

This means that agent rather than preventing  $A_A$  from fulfilling its duties, a policy can be written where  $A_A$  can be given access to pieces of medical data. The system also continues to prevent  $A_T$  from accessing medical data, as required.

This illustrates that the architecture of a system can be used to design a more effective policy than one based solely on clearance levels. In addition, the system model employed here allows policies applying to an individual agent (or group of agents) to be developed.  $\square$

**Example 4.7** (Composing Bell-LaPadula and Biba Policy Models). An important and subtle problem concerns the construction of systems that support both BLP confidentiality and Biba integrity models. A direct use of the same notion of security level for both BLP and Biba policies (with no additional refinements) evidently leads to systems that are somewhat trivial, in the sense that data subjected to both policies never flows from one level to any other.

A more subtle combination is the *dual implementation* method for composing the BLP and Biba models. This involves restricting access of an agent or data item based on the rules of BLP or Biba, but not both [3].

In the dual implementation, agents are either confidentiality agents (intended to observe the BLP policy) or integrity agents (intended to observe the Biba) policy. The set of agents  $\mathcal{A}$  is thus partitioned into sets  $\mathcal{A}_C$  and  $\mathcal{A}_I$ .

Data, on the other-hand, is *not* partitioned. Both the BLP and Biba policies apply to all data. An agent is  $\mathcal{A}_C$  thus satisfies NRU and NWD for this data, whilst an agent in  $\mathcal{A}_I$  satisfies NRD and NWU for this data.

Consider the example indicated in Figure 12. This consists of a conjoined pair of security systems. The left-hand security system is intended to maintain a BLP policy. The right-hand system is intended to maintain a Biba policy. Each system has two levels, marked 1 and 2. In the BLP system, 1 is intended to be high confidentiality and 2 low confidentiality. Assume that all the vertices and edges of the conjoined system are shown in the figure. All of the confidentiality agents are located in the left-hand system, and all of the integrity agents in the right-hand system. In particular, an agent  $A_1 \in \mathcal{A}_C$  lives in level 1 of the left-hand system, an agent  $A_2 \in \mathcal{A}_I$  lives in level 1 of the right-hand system, and an agent  $A_3 \in \mathcal{A}_3$  lives in level 2 of the right-hand system at the locations shown. A data item  $D$  lives at the same location as the agent  $A_1$ . It is then immediately evident that  $A_1$  can write  $D$  across to level 2 on the right-hand system. Moreover, if  $D$  can reach the location of  $A_3$  then it could be written back across to level 2 of the left-hand system. At such a point, information would have been written down from high-confidentiality to low-confidentiality, violating the intended BLP policy. Consider two cases, where the two systems are aligned or anti-aligned.

First, suppose that the levels of the two systems are aligned: level 1 on the right is high integrity and level 2 is low integrity. In this case, agent  $A_2$  can write data ‘down’ on the right to the location of  $A_3$ . The BLP policy is thus violated through a sequence of actions.

Now suppose that the two systems are anti-aligned: level 1 on the right is low integrity, and level 2 is high-integrity. In this case,  $A_2$  cannot write the data ‘up’ to level 2, and the BLP

policy is retained. Moreover, no confidentiality agent can ever write data from level 1 to the high-integrity level 2. This can give some resistance against ‘Trojan Horse’ attacks that attempt to use ordinary user accounts on machines to compromise system files that should only be modified by administrators [3].

The situation becomes much more complex when there are more complex systems of levels.

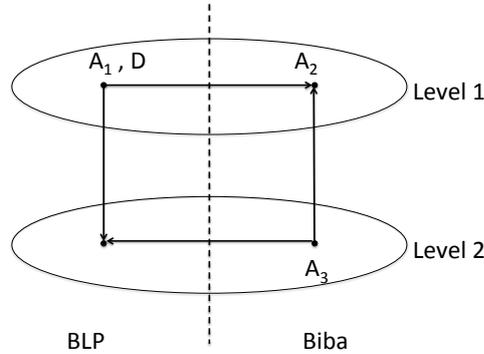


FIGURE 12. A Dual Implementation Problem

□

**Example 4.8** (Cascade Vulnerabilities). We begin with a brief summary of the problem, following the description in [44, 36, 8]. A network can be given a security accreditation (rating) based on assurance levels, which impacts the data it can contain, and moreover: ‘These accreditation ranges are taken into account when determining whether or not a component should be allowed to connect to the system. In this way, the potential damage that can occur when information is compromised or modified can be limited to an acceptable level’ [44].

The cascading problem is identified in [44]: ‘The cascading problem exists when a penetrator can take advantage of network connections to compromise information across a range of security levels that is greater than the accreditation range of any of the component systems he must defeat to do so.’ In straightforward terms, the transitivity of connections can lead to the composability of vulnerabilities. A composite vulnerability can be more serious than its components, and accreditation ratings are closely related to the potential for vulnerabilities to cause undesirable information flows between security levels.

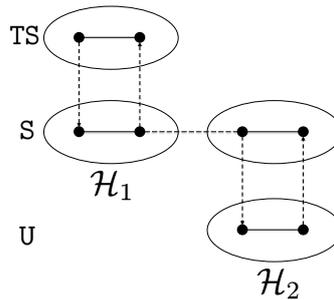


FIGURE 13. (ability)

As an example, consider Figure 13. The network  $\mathcal{H}_1$  has only Top Secret, TS and Secret, S, levels. The network  $\mathcal{H}_2$  has only Secret and Unclassified, U, levels. The two networks  $\mathcal{H}_1$  and  $\mathcal{H}_2$  have been joined by a new connection at the Secret level. An attacker who could defeat a BLP system in  $\mathcal{H}_1$  could cause Top Secret information to flow down to the Secret level: the severity of this potential corresponds to a certain accreditation rating. An attacker who could defeat a BLP system in the  $\mathcal{H}_2$  network could cause information to flow from Secret to Unclassified: the severity of this potential corresponds to another accreditation rating. However, an attacker who could defeat the BLP systems of both  $\mathcal{H}_1$  and  $\mathcal{H}_2$  in the conjoined system, could cause information to

flow from Top Secret to Unclassified: this potential is more severe than that allowed under either of the two previous accreditations. This example illustrates that accreditation and cascading can be understood in terms of layered graph models.  $\square$

**4.2. Language Extensions for Particular Models.** The language is intended to be used to describe real systems example in a degree of detail. Sometimes the core language set-out above is insufficient alone for this purpose. For particular examples it can be useful to add information to the language pertaining to those particular models. We have found certain language extensions of particular kinds to be useful.

Given a stack  $G = G_1 \hat{\underline{\textcircled{a}}} \dots \hat{\underline{\textcircled{a}}} G_n$ , the language is extended as follows:

$$q ::= p \mid A \mapsto^a D \mid A \mapsto_{\text{U}}^a D \mid A \mapsto_{\text{D}}^a D, \mid A \mapsto_{G_i}^a D \mid A \mapsto_{\Theta}^a D,$$

where  $G_i$  is one of the layers of the stack and  $\Theta$  is an expression relating to the stack structure as further explained below. The semantics of these assertions is encapsulated in Table 7.

A useful access relation that describes privilege within a specific layer,  $G_i$ , is defined by

$$(5) \quad F, G \models A \mapsto_{G_i}^a D \quad \text{iff} \quad \begin{array}{l} \text{there is a path } P = F(A), \dots, F(D) \text{ in } G \\ \text{such that } V(P) \subseteq V(G_i) \text{ and,} \\ (A, \mathbf{a}, D) \in \text{Label}(e) \text{ for all } e \in E(P). \end{array}$$

The satisfaction of this relation means that the path from  $A$  to  $D$  is contained entirely in  $G_i$ , and that the edges of the path are all labelled with the appropriate access permission.

A *path property*,  $\Theta$ , is a formula of some logic of paths. For example, for our purposes here, it is enough to have a first-order language over two sorts, with one sort for the vertices and the other for the edges of the ambient graph, constants to name every vertex and edge, and predicate symbols to name every layer and distinguished edge set. Path properties could be formalized further but this is not necessary for the purposes of this presentation.

Definiens 5, 2 and 3 are specific cases of the definiens of a general access assertion,  $A \mapsto_{\Theta}^a D$ , given as follows:

$$(6) \quad F, G \models A \mapsto_{\Theta}^a D \quad \text{iff} \quad \begin{array}{l} \text{there is a path } P = F(A), \dots, F(D) \text{ in } G \\ \text{such that the path property } \Theta \text{ holds of } P \text{ and,} \\ (A, \mathbf{a}, D) \in \text{Label}(e) \text{ for all } e \in E(P). \end{array}$$

Satisfaction of this general access assertion requires that a path, satisfying some path property, exists within the vertices of a stack  $G$ , and that such a path is labelled with the relevant permissions. The relations shown in 5, 2 and 3 enforce conditions about the direction of a path through  $G$ .

---


$$F, G \models A \mapsto_{G_i}^a D \quad \text{iff} \quad \begin{array}{l} \text{there is a path } P = F(A), \dots, F(D) \text{ in } G \\ \text{such that } V(P) \subseteq V(G_i) \text{ and,} \\ (A, \mathbf{a}, D) \in \text{Label}(e) \text{ for all } e \in E(P). \end{array}$$

$$F, G \models A \mapsto_{\Theta}^a D \quad \text{iff} \quad \begin{array}{l} \text{there is a path } P = F(A), \dots, F(D) \text{ in } G \\ \text{such that the path property } \Theta \text{ holds of } P \text{ and,} \\ (A, \mathbf{a}, D) \in \text{Label}(e) \text{ for all } e \in E(P). \end{array}$$

TABLE 7. Satisfaction Relation for Language Extensions

---

**Example 4.9** (A Confidential Filestore). In [3], a BLP compliant system is illustrated using a simple example about access to files in a repository. We present a similar example here.

Let  $G = G_1 \hat{\underline{\textcircled{a}}} G_2$  be a bi-layered stack representing the network of an organization, and let  $\mathcal{R}$  denote a data repository. This repository is represented as a subset of  $V(G_1)$ . We assume if data is mapped to a vertex of  $G_1$  (the high-confidentiality layer), the vertex is contained in  $\mathcal{R}$ . All data resident in  $\mathcal{R}$  has the clearance level 1. The system supports two actions which, when successful, allow an agent to look inside (**read to**)  $\mathcal{R}$  or to append (**write to**)  $\mathcal{R}$ .

Following the presentation in [3], we show that the BLP rules hold relative to  $\mathcal{R}$ . Assume that the organization specifies that a **read** action can only be carried out by an agent of equal clearance level to  $\mathcal{R}$ , while there are no restrictions placed on **write**. This specification means that,

$$F, G \models A \mapsto_{G_1}^r D$$

using the single-layer access assertion of Table 7. The fact that access to data in  $\mathcal{R}$  is compliant with the no read-up (NRU) rule is captured by this instance of the satisfaction relation.

Given that there are no restrictions on agents writing to data in  $\mathcal{R}$ ,

$$F, G \models A \mapsto_{\top}^w D.$$

with  $\mapsto_{\top}^w$  an instance of the general access assertion  $\mapsto_{\mathcal{G}}^a$  as in Table 7. Access to data in  $\mathcal{R}$  is trivially compliant with the no write-down (NWD) rule.  $\square$

**Example 4.10 (UNIX).** This example uses our model to represent access permissions in a simple UNIX-like system ([www.unix.org](http://www.unix.org)). It is not intended to be a fully general description of the permission system of any UNIX variant.

Suppose that the system permissions have all been fixed in advance. Consider a user  $A$ . Suppose that  $A$  is the owner of a directory  $D$ . Let  $\text{Group}(D)$  denote the group associated with  $D$ . For simplicity, assume that  $A$  is a member of  $\text{Group}(D)$ . Suppose that  $A$  specifies that write access to  $D$  is available to  $A$  only, while members of  $\text{Group}(D)$  have read access. No access is permitted to non-group members. In UNIX, this would be represented as

$$\text{dir rw - r - - - - } A \text{ Group}(D).$$

We assume that we have a UNIX directory that can be regarded as a tree (and not a directed graph, as can occur under some implementations), with a subdirectory (or file) belonging to a directory being the child-parent relationship. Consider a particular such tree,  $T$ , and add a formal inverse to every directed edge. The resulting graph,  $G$ , can now be regarded as a bi-layered graph in more than one way.

Since our logical models are presently restricted to stacks we make a simplistic choice that each layer consists of all the nodes at each tree height. The set-up is depicted in Figure 14.

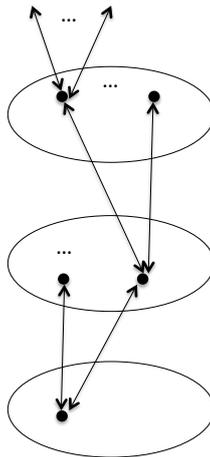


FIGURE 14. A Simple UNIX File System

Let the specified system permissions generate the appropriate ACLs (in the sense used in this paper, not in the UNIX sense) on the edges of  $G$ . Let  $S$  be an operating system subject (process). The execution of this (invoked by some user) is associated with a particular point in the directory tree (the directory that the user calls it from). Then, in this example, the access clauses can be stated using group membership. If each agent adheres to the system policies, access conditions to

$D$  can be specified by the following judgements:

$F, G \vDash S \mapsto_{\text{UX}}^r D$  iff there is a path  $P = F(S), \dots, F(D)$  in  $G$   
 and  $S$  is executing on behalf of and with the permissions of some user  $B$   
 and the final edge of  $P$  (into  $F(D)$ ) is labelled with  $(B, \mathbf{r}, D)$   
 and all other edges of  $P$  are labelled with  $(B, \mathbf{x}, D)$   
 and  $B \in \text{Group}(D)$

and

$F, G \vDash S \mapsto_{\text{UX}}^w D$  iff there is a path  $P = F(S), \dots, F(D)$  in  $G$   
 and  $S$  is executing on behalf of and with the permissions of the user  $A$   
 and the final edge of  $P$  (into  $F(D)$ ) is labelled with  $(A, \mathbf{w}, D)$   
 and all other edges of  $P$  are labelled with  $(A, \mathbf{x}, D)$ .

Only execute,  $\mathbf{x}$ , permission is required to traverse intermediate directories (between that of  $S$  and that of  $D$ ) for all access operations.

The relations  $\mapsto_{\emptyset}^a$  are not quite sufficiently general to capture these two access relations. However the similarity is clear. We do not discuss the many further nuances of permissions for real UNIX-like operating systems.  $\square$

The example above employs a very simple version of the access control set-up found in UNIX [49]. To handle all of the structure and complexity of the full UNIX access control régime, and lattice models more generally, would require a much richer logical set-up, for which there are several design choices to be made. For example, in order to capture the full range of access permissions that are assignable in UNIX — see [3] for a relevant discussion — it would seem to be necessary to work not with stacks, but rather with the more general layered graph models of **LGL**. Such a choice would require the access relations to be formulated in this more general setting, taking account of the weaker algebraic and combinatorial structure of the more general models. Such a development, which we believe should also take account of a range of examples other than UNIX, represents a future project beyond our present scope.

In subsequent sections, where a layer-specific assertion  $A \mapsto_{G_i}^a D$  or a general path access assertion  $A \mapsto_{\emptyset}^a D$  is used, it is to be understood that an appropriate language extension for a particular model is under consideration.

## 5. CHINESE WALLS: ADDING $*$ (AND $\rightarrow$ )

As described so far, the use of multiplicative connectives in **LGL** is restricted to non-commutative layering. It is also possible to add, as in BI and Separation Logic, a commutative and associative multiplicative conjunction,  $*$ , and its associated implication,  $\rightarrow$ . Such an addition is useful for describing an important class of access control policy models sometimes known as Chinese Walls [3]. These models decompose a system into regions between which information is not permitted to flow. Table 8 gives the rules required to extend the Hilbert-type proof system given in Table 1 with  $*$  and  $\rightarrow$ .

$$\frac{\xi \vdash \phi \quad \eta \vdash \psi}{\xi * \eta \vdash \phi * \psi} \quad \frac{\eta * \phi \vdash \psi}{\eta \vdash \phi \rightarrow \psi} \quad \frac{\xi \vdash \phi \rightarrow \psi \quad \eta \vdash \phi}{\xi * \eta \vdash \psi}$$

TABLE 8. Additional Proof Rules for  $*$  and  $\rightarrow$

**Definition 5.1.** Let  $\mathcal{G}$  be an ambient graph. Define a partial binary operation  $\circ$  on the set of subgraphs of  $\mathcal{G}$ . For  $G_1, G_2 \sqsubseteq \mathcal{G}$ ,  $G_1 \circ G_2$  is defined iff  $V(G_1) \cap V(G_2) = \emptyset$ , and there is no edge of  $\mathcal{G}$  from a vertex of  $G_1$  to a vertex of  $G_2$ , and no edge of  $\mathcal{G}$  from a vertex of  $G_2$  to a vertex of  $G_1$ .

When defined, the result of the operation  $\circ$  is the union of disjoint subgraphs of  $\mathcal{G}$ . This operation is both commutative and associative. We work with the binary case here, but it would be easy to define an  $n$ -ary version. The additional semantic clauses for satisfaction are shown in Table 9. This set up requires Definition 5.2, which is similar to Definition 4.1.

**Definition 5.2.** Let  $\mathcal{G}$  be an ambient graph and let  $\mathbb{F}$  be defined as in Definition 4.1. Define a partial, binary operation  $\circ : \mathbb{F} \times \mathbb{F} \rightarrow \mathbb{F}$  in the following way. Let  $F_1 \circ F_2$  be defined if and only if there exists  $G_1, G_2 \in \mathbf{Sg}(\mathcal{G})$  such that  $G_1 \circ G_2$  is defined, the image of  $F_1$  is contained in  $V(G_1)$ , the image of  $F_2$  is contained in  $V(G_2)$  and the domains of  $F_1$  and  $F_2$  are disjoint. When defined,  $F_1 \circ F_2$  produces a function with image contained in  $V(G_1 \circ G_2)$ .

In Section 4, the interpretation was of the form  $F, G \models \phi$ , where  $G$  was a graph, formed as a specific stack. In this section, the graph can now be formed from both of the operations  $\circ$  and  $\hat{\circ}$ . As a simplifying measure, we consider only a single pair of distinguished edge sets  $(\mathcal{E}, \mathcal{F})$  to allow us to forget about the composite structure and take  $G$  simply to be the graph. The definiens given for the logical atoms given in the relations 5, 2, 3, 6 can then understood unambiguously in this setting. The interpretation set out in Section 4 is extended to the new language including the connectives  $*$  and  $\rightarrow$  as shown in Table 9.

---

$F, G \models \phi_1 * \phi_2$	iff for some $F_1, G_1, F_2, G_2$ such that $F = F_1 \circ F_2$ and $G = G_1 \circ G_2$ , $F_1, G_1 \models \phi_1$ and $F_2, G_2 \models \phi_2$
$F, G \models \phi \rightarrow \psi$	iff for all $J, H$ such that $F \circ J \downarrow$ and $G \circ H \downarrow$ , $J, H \models \phi$ implies $F \circ J, G \circ H \models \psi$

---

TABLE 9. Additional Satisfaction Clauses

**Example 5.3** (Multi-layer Chinese Walls). As usually formulated, the Chinese Walls model does not deal with the structure of multi-layered systems. If poorly formulated, or implemented, in the presence of multiple layers, the rules of a policy that uses Chinese Walls may then not guard against unwanted potential flows of information through the network. This example shows how such violations can occur and how flaws can be corrected through the composition of Chinese Walls with the rules of either BLP or Biba.

Consider an organization that adopts a Chinese Wall model to segregate access in its physical information network. Suppose that the organization also employs agents as consultants who, working outside the organizations premises, are provided with a laptop to access data relating to the organization. This results in an additional layer of organizational structure, which we will call the Mobile Access Layer.

Consider Figure 15. This shows graphs  $G_1$  and  $G_2$  at either sides of a Chinese Wall, layered over another graph  $G_3$ . The vertices and edges of the ambient graph,  $\mathcal{G}$ , are precisely those represented in this figure. The situation shows that a Chinese Wall could be circumvented because of the presence of the lower layer. The intra-layer edges each represent a pair of edges, with one in each direction. Each of the inter-layer edges is marked with the distinguished edge set to which it belongs.

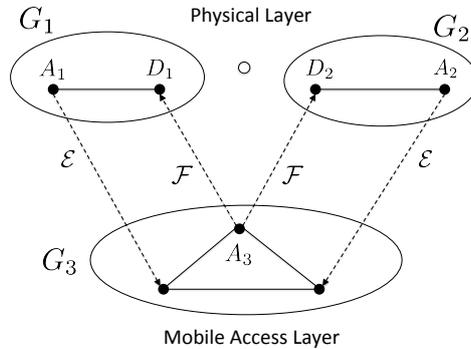


FIGURE 15. Chinese Wall in a Layered Graph

Let  $\bar{G}$  denote the Physical Layer,  $\bar{G} = G_1 \circ G_2$ . Let  $G_3$  denote the Mobile Access Layer, let  $G = \bar{G} \hat{\otimes} G_3$ .

Let  $A_1$  and  $A_2$  be agents at a firm (such as an investment bank advising on a merger or an acquisition), segregated to work solely on datasets  $D_1$  and  $D_2$ . Let  $A_3$  be an agent at a vertex of  $G_3$  capable of accessing  $V(G_1)$  and  $V(G_2)$ , (say through the use of a laptop, as described above). Assume that  $F_1$  has domain  $\{A_1, D_1\}$ ,  $F_2$  has domain  $\{A_2, D_2\}$  and  $F_3$  has domain  $\{A_3\}$ . Assume that these together map the elements of these domains to the vertices of  $G$  as shown in Figure 15. Let  $\bar{F} = F_1 \circ F_2$  and let  $F = \bar{F} \circ F_3$ .

Assume that all edges of  $G$  are labelled with the relevant ACLs so that each  $A_i$  can read or write any data item  $D_j$  when their locations are connected by an edge.

Then, if  $\bar{G}$  is considered as a stand-alone network, that is, forgetting about  $G_3$ , the same access relations between the agents and data items located in  $\bar{G}$  hold. We say that *segregated access* (between  $G_1$  and  $G_2$ ) is maintained; this is represented by the following judgement

$$\bar{F}, \bar{G} \models ((A_1 \mapsto_{G_1}^a D_1) * (A_2 \mapsto_{G_2}^a D_2))$$

where the instance of the  $*$ -connective can be witnessed by the decomposition  $\bar{G} = G_1 \circ G_2$ . Let  $\theta$  be  $((A_1 \mapsto_{G_1}^a D_1) * (A_2 \mapsto_{G_2}^a D_2))$ .

The existence of the Mobile Access Layer means that data can flow from  $G_1$  to  $G_2$  as there are presently no rules governing access across layers. For  $A_3$  in  $G_3$  it is possible, through a combination of read and write actions, to violate the Chinese Wall. The agent  $A_3$  could read upwards to  $D_1$  and subsequently write this information upwards to the location of  $D_2$ , meaning that  $A_2$  can gain access to the data contained in  $D_1$ . A properly implemented Chinese Wall should ensure that it can never be the case that  $A_2$  can access  $D_1$ , or for  $A_1$  to access  $D_2$ . It is therefore necessary to implement additional policies to ensure that unwanted access does not occur. This can be done by modifying the ACLs on the edges of  $\mathcal{E}$  and  $\mathcal{F}$ .

The organization can choose to implement the rules of either BLP or Biba, depending on whether confidentiality or integrity is its main concern. If the choice is to implement BLP then the initial read-up action of  $A_3$  would be prevented by the NRU rule. The implementation of Biba's rules would prevent the final write-up action of  $A_3$  through the NWU requirement.

Let  $\phi$  be a correctness condition that means that  $A_3$  is located in  $G_3$ , so that  $F_3, G_3 \models \phi$ . Then the resolution of the potential flaw is represented by the following properties:

$$\begin{aligned} \text{BLP (NRU)} : \quad & F, G \models (\theta \blacktriangleright \phi) \wedge \neg(A_3 \mapsto_{\bar{G}}^r D_1) \wedge \neg(A_3 \mapsto_{\bar{G}}^w D_2) \\ \text{Biba (NWD)} : \quad & F, G \models (\theta \blacktriangleright \phi) \wedge \neg(A_3 \mapsto_{\bar{G}}^w D_1) \wedge \neg(A_3 \mapsto_{\bar{G}}^r D_2). \end{aligned}$$

So, the use of either the BLP or Biba model will prevent circumvention by prohibiting the initial or final action. Given that the rules relate to specific data items, agents mapped to  $G_3$  can retain access to other data items. BLP will also prevent a malicious user, for example  $A_1$  in  $\bar{G}$ , from writing data to  $G_3$ , where it could be read by  $A_2$ . A similar result will hold for Biba if the layered set-up is  $G_3 \hat{\otimes} \bar{G}$ .  $\square$

## 6. DYNAMICS AND THE FRAME RULE

The logic **LGL** can be extended with a simple form of action and modality [21], along the lines of Hennessy–Milner logic [34] or dynamic logic (e.g., [33]), in order to provide a mechanism for describing how a system may evolve.

One way in which a system may evolve is with a change to the underlying architecture, with connections being added or deleted, so further composing or decomposing the system. The addition or removal of links allows the presentation of situations where an organization chooses to restrict access by removing a link, or situations involving access across a connection that is not permanently open. In graph models, this can be represented by changing the edges of a graph. As in the previous sections, we work with graphs that have at most a single edge in any direction between any pair of vertices. As in Section 5, we consider only a single pair of distinguished edge sets  $\mathcal{E}, \mathcal{F}$ .

Let  $\mathbf{add}(v, v')$  and  $\mathbf{remove}(v, v')$  be actions that add and remove an edge from vertex  $v$  to vertex  $v'$  within a graph. We exclude actions of this kind for edges of the distinguished edge sets,  $\mathcal{E}, \mathcal{F}$ . In particular, inter-layer edges will not be changed by actions. Let  $\mathbf{b}$  range over the actions of the forms  $\mathbf{add}(v, v')$  and  $\mathbf{remove}(v, v')$ .

An **add** action may alter the ambient graph structure from  $\mathcal{G}$  to some  $\mathcal{G}'$  by adding an edge not present in  $\mathcal{G}$ ; the **remove** action is also assumed to remove the edge from the ambient graph, as well as the sub-graph under consideration. Let  $(\mathcal{G}, G) \xrightarrow{\mathbf{b}} (\mathcal{G}', G')$  denote that a given graph  $G$  evolves to  $G'$  via the action  $\mathbf{b}$ , and that the ambient graph evolves from  $\mathcal{G}$  to  $\mathcal{G}'$ .

The logic we now introduce extends that of Section 5. For simplicity here, we assume the only atomic formulae are access assertions. In addition, actions that reconfigure graphs generate action modalities. We give the satisfaction clauses for these modalities below before giving two illustrative examples (restoring the explicit ambient graph in the notation):

$$(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F, G \models \ll \mathbf{b} \gg \phi \quad \text{iff} \quad \begin{array}{l} \text{there exist } \mathcal{G}', G' \text{ such that } (\mathcal{G}, G) \xrightarrow{\mathbf{b}} (\mathcal{G}', G'), \\ \text{and } (\mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle), F, G' \models \phi \end{array}$$

$$(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F, G \models \|\mathbf{b}\| \phi \quad \text{iff} \quad \begin{array}{l} \text{for all } \mathcal{G}', G' \text{ such that } (\mathcal{G}, G) \xrightarrow{\mathbf{b}} (\mathcal{G}', G'), \\ \mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle, F, G' \models \phi. \end{array}$$

**Example 6.1** (Guarded Access). Consider an organization that keeps sensitive data in a secure location. To access this data, agents must satisfy an ID check. If an agent satisfies the check, a link is added from the location of the agent to the location of the data to allow the access. If it is possible that another agent might be able to piggyback on a successful ID check (such as by tailgating through a barrier) then an edge from the second agent to the first will be deleted to prevent this. This situation is shown in Figure 16, where agents  $A$  and  $B$  are unable to access a

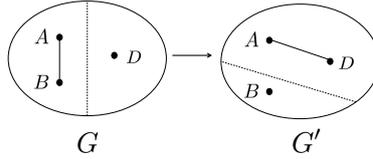


FIGURE 16. Guarded Access

data item  $D$ . The second part of Figure 16 shows that  $A$  has successfully identified itself and been granted access to  $D$ , while  $B$  still cannot access  $D$  (via a path in the graph under consideration) because of the reconfiguration.

Let  $\mathcal{G} = G$  be the graph on the left of Figure 16, where all edges marked represent an edge in both directions. Let  $F$  be the associated agent-data assignment that maps agents and data to the vertices shown.

Let **open** be the action that adds the edge between  $A$  and  $D$  and let **close** be the action that removes the link between  $A$  and  $B$ . Note that these are merely instances of the **add** and **remove** actions introduced above. Let  $\mathcal{G}' = G'$  denote the graph after these reconfiguration actions have taken place. Then the action modalities defined above give the access condition:

$$(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F, G \models \ll \mathbf{open} \gg \ll \mathbf{close} \gg ((A \mapsto_{\mathcal{G}'}^a D) \wedge \neg(B \mapsto_{\mathcal{G}'}^a D))$$

This property shows that the network will allow  $A$  access to  $D$ , after the appropriate reconfigurations have taken place, while  $B$  is still unable to access  $D$  (this is, of course, contingent upon  $A$  having the necessary credentials).  $\square$

**Example 6.2** (Establishment of a Chinese Wall). This example is similar to Example 5.3 and describes the establishment of a Chinese Wall. Consider a situation (again, such as in the firm advising on a merger or an acquisition) in which agents are able to choose which data items they access. Upon accessing certain data items, agents may be precluded from accessing others if they contain data deemed to conflict (so rendering their advice on the deal tainted).

The left-hand side of Figure 17 shows an agent  $A_1$  who could potentially access documents  $D_1$  or  $D_2$  while  $A_2$  is assumed to have access to  $D_2$  only. The right-hand side of Figure 17 shows the construction of a Chinese Wall to prevent  $A_1$  from accessing  $D_2$  after choosing to access  $D_1$ . Let the action of removing the link as shown be denoted as **wall**. This is just a particular instance of a **remove** action. Let  $\mathcal{G} = G$  be the network shown on the left-hand side of Figure 17. Let  $F$  be the evident agent-data assignment mapping the agents and data to vertices of  $G$  as shown in the

figure. Let  $\mathcal{G}' = G' = G_1 \circ G_2$  be the graph as shown on the right-hand side of Figure 17, where  $G_1$  is the component containing  $A_1, D_1$ , and  $G_2$  is the component containing  $A_2, D_2$ . The graph  $G'$  is the network after the **wall** action. In both the left-hand side and the right-hand side, all of the undirected edges represent a pair of directed edges, one in each direction.

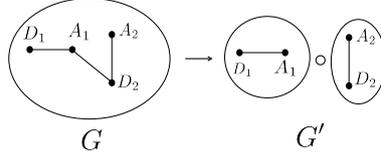


FIGURE 17. Establishment of a Chinese Wall

The following relations describe that, in this example  $A_1$  can read  $D_1$ , and  $A_2$  can read  $D_2$  both before and after the **wall** action, and  $A_1$  cannot read  $D_2$  and  $A_2$  cannot read  $D_1$  after the **wall** action:

$$(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F, G \models (A_1 \mapsto_{\mathcal{G}}^r D_1) \wedge (A_1 \mapsto_{\mathcal{G}}^r D_2) \wedge (A_2 \mapsto_{\mathcal{G}}^r D_2)$$

$$(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F, G \models \ll \mathbf{wall} \gg ((A_1 \mapsto_{G_1}^r D_1) * (A_2 \mapsto_{G_2}^r D_2)).$$

□

**6.1. Intra-layer Frame Rule.** The reconfiguration of graphs (i.e. changes in system architecture) can have an adverse effect on the implementation of access control policies. To be able to reason about system structures where changes in the architecture do not impact security measures, we use a frame rule, similar to the rule adopted in Pointer Logic [37]. Our version of the frame rule is defined as follows:

**Definition 6.3** (Intra-layer Frame Rule). Let  $\mathcal{G}$  be an ambient graph and let  $G, H$  be subgraphs of  $\mathcal{G}$  such that  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F, G \models \phi$  and  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), J, H \models \psi$ . Let  $\mathbf{b}$  be an action such that  $(\mathcal{G}, G) \xrightarrow{\mathbf{b}} (\mathcal{G}', G')$  and let  $(\mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle), F, G' \models \phi'$ . Suppose that the action  $\mathbf{b}$  affects the edges of  $G$  *only* and that  $F \circ J, G \circ H$  are both defined. Then:

$$(\mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle), F \circ J, G' \circ H \models \phi' * \psi$$

holds.

Suppressing the auxiliary data when it is contextually clear, we write the frame rule as

$$\frac{\phi \quad \mathbf{b} \quad \phi'}{\phi * \psi \quad \mathbf{b} \quad \phi' * \psi}.$$

Although we call it a ‘rule’ it is properly a relation between instances of the satisfaction relation. The fact that this relation holds under all interpretations is more-or-less immediate, noting that  $G' \circ H$  is defined, and so that  $G'$  and  $H$  give the required witnesses for  $\phi' * \psi$ .

The following is obtained straightforwardly:

**Proposition 6.4.** *The intra-layer frame rule is valid.*

**Example 6.5** (Intra-layer Frame Rule). This example extends Example 6.1 to give a simple illustration of the intra-layer frame rule. Consider an organization that uses some form of security check to guard access, as described earlier. Suppose now that in addition to this security implementation, a Chinese Wall is in place to maintain complete segregation of particular items of data.

This situation is shown in Figure 18, which extends Figure 16 to show an agent  $A'$  with access to  $D_2$  while segregated from the area in which guarded access takes place. Let  $G_1, G_2$  be as indicated on the left-hand side of Figure 18, where the ambient graph is  $\mathcal{G} = G_1 \circ G_2$ . The undirected edges marked represent pairs of directed edges, one in each direction. Let  $F_1, F_2$  be

the evident agent-data assignments mapping the agents and data into  $G_1$  and  $G_2$ , respectively, as indicated in the figure. Let  $\phi$  be a correctness condition associated with  $A$  being located in  $G_1$ . Let  $\phi'$  be  $(A \mapsto_{G_1}^a D_1) \wedge \neg(B \mapsto_{G_1}^a D_1)$  and let  $\psi$  be  $A' \mapsto_{G_2}^a D_2$ . For simplicity, let **change** denote the combined actions of adding and removing the edges as shown. Since **change** only

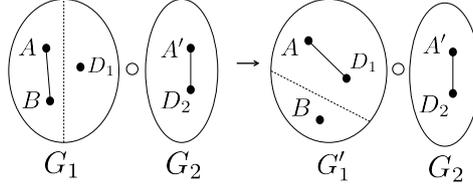


FIGURE 18. Guarded Access and a Chinese Wall

affects the structure of  $G_1$  and it is clear that  $G_1 \circ G_2$  is defined, then  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F_1, G_1 \models \phi$ , and  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F_2, G_2 \models \psi$ , and  $(\mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle), F_1, G_1' \models \phi'$ , and  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F_1 \circ F_2, G_1 \circ G_2 \models \phi * \psi$ .

The premisses for the intra-layer frame rule are therefore satisfied, and we may conclude that  $(\mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle), F_1 \circ F_2, G_1' \circ G_2 \models \phi' * \psi$ . The satisfaction of the frame rule in this example shows that the Chinese Wall, which segregates the access of the agents, is unaffected by the change in graph structure resulting from an agent's successfully providing ID to access data.  $\square$

**6.2. Inter-layer Frame Rule.** The reconfiguration of edges within a single graph can have an effect on other layers in a multi-layer system. It is possible that a change in one layer could unintentionally lead to some policy being undermined, or introduce a requirement to check that a policy still holds. For this reason, it is necessary to also develop an inter-layer frame rule. In the remainder of this section, frame rules and examples are stated using the binary versions of graph composition and the multiplicative connector  $\blacktriangleright$ .

**Definition 6.6** (Inter-layer Frame Rule 1). Let  $\mathcal{G}$  be an ambient graph and let  $G, H$  be subgraphs of  $\mathcal{G}$  such that  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F, G \models \phi$  and  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), J, H \models \psi$ . Let  $\mathbf{b}$  denote an action such that  $(\mathcal{G}, G) \xrightarrow{\mathbf{b}} (\mathcal{G}', G')$  and let  $(\mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle), F, G' \models \phi'$ . Suppose the action  $\mathbf{b}$  affects the edges of  $G$  *only* and that  $F \hat{\circ} J, G \hat{\circ} H$  are both defined. Then

$$(\mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle), F \hat{\circ} J, G' \hat{\circ} H \models \phi' \blacktriangleright \psi$$

holds.

**Definition 6.7** (Inter-layer Frame Rule 2). Let  $\mathcal{G}$  be an ambient graph and let  $G, H$  be subgraphs of  $\mathcal{G}$  such that  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), F, G \models \phi$  and  $(\mathcal{G}, \langle \mathcal{E}, \mathcal{F} \rangle), J, H \models \psi$ . Let  $\mathbf{b}$  denote an action such that  $(\mathcal{G}, G) \xrightarrow{\mathbf{b}} (\mathcal{G}', G')$  and let  $(\mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle), F, G' \models \phi'$ . Suppose the action  $\mathbf{b}$  affects the edges of  $G$  *only* and that  $J \hat{\circ} F, H \hat{\circ} G$  are both defined. Then

$$(\mathcal{G}', \langle \mathcal{E}, \mathcal{F} \rangle), J \hat{\circ} F, H \hat{\circ} G' \models \psi \blacktriangleright \phi'$$

holds.

Again, suppressing the auxiliary data, we write the two frame rules as

$$\frac{\phi \quad \mathbf{b} \quad \phi'}{\phi \blacktriangleright \psi \quad \mathbf{b} \quad \phi' \blacktriangleright \psi} \quad \text{and} \quad \frac{\phi \quad \mathbf{b} \quad \phi'}{\psi \blacktriangleright \phi \quad \mathbf{b} \quad \psi \blacktriangleright \phi'}.$$

Again, these are abbreviations for relations between instances of the satisfaction relation. The fact that these relations holds for any interpretation follows using the  $G'$  and  $H$  that appear in the premisses to witness the satisfaction of  $\phi' \blacktriangleright \psi$  for the first rule, or  $\psi \blacktriangleright \phi'$  for the second rule.

The following is obtained straightforwardly:

**Proposition 6.8.** *The inter-layer frame rules are valid.*

**Example 6.9** (Inter-layer Frame Rule). This example adapts and extends Examples 5.3 and 6.2 to present an instance of the inter-layer frame rule. As in Example 6.2, consider a firm, such as an investment bank, advising on a merger or acquisition. The firm’s policy is that, subsequent to an employee accessing a data item that is in conflict with other data within the deal, a suitable Chinese Wall will be implemented. Moreover, suppose that the firm’s organizational structure is multi-layered, with an access control policy model regulating information flow between layers. The firm must ensure that the Chinese Wall is not undermined by this layered structures. Now consider Figure 19. The graph on the left-hand side is the ambient graph  $\mathcal{G}$  with distinguished edge sets  $\mathcal{E}, \mathcal{F}$ . Let  $G = \mathcal{G}$ . The graph on the right-hand side is  $\mathcal{G}' = G'$ . The undirected edges represent pairs of directed edges, one in each direction. The two-headed dotted arrows represent a pair of arrows, one downward and in the distinguished edge set  $\mathcal{E}$ , and one upward and in the distinguished edge set  $\mathcal{F}$ . Moreover,  $G = G_1 \hat{\circ} G_2$  and  $G' = G'_1 \hat{\circ} G_2$ . Let  $F_1$  and  $F_2$  be the agent-data assignments that map agents and data to the vertices of  $G_1$  and  $G_2$ , respectively, as indicated in the figure.

The organization must be confident that, in the presence of  $G_2$ , the Chinese Wall that has been constructed through the evolution of  $G_1$  (from the left-hand side to the right-hand side of the figure) is secure and cannot be circumvented. As discussed in Example 5.3, circumvention could

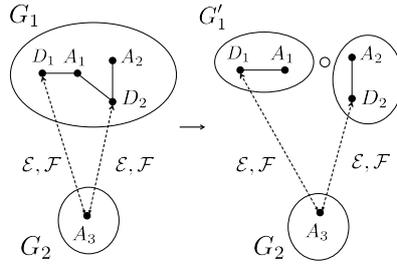


FIGURE 19. Formation of Chinese Wall in Multi-layer Environment

occur as a result of a combination of read and write actions by agent  $A_3$  and is prevented by the composition of Chinese Walls and the policies of either BLP or Biba.

Assume  $A_1$  has access to  $D_1$  and  $A_2$  has access to  $D_2$  on the left-hand side of the figure. Let  $\xi$  be valued so that it simply requires that  $A_3$  is located in  $G_2$ . Then  $F_2, G_2 \models \xi$  (for ambient graph  $\mathcal{G}$ ). Let  $\phi$  be the property  $(A_1 \mapsto_{G_1}^a D_1) \wedge (A_2 \mapsto_{G_1}^a D_2) \wedge (A_1 \mapsto_{G_1}^a D_2) \wedge (A_2 \mapsto_{G_1}^a D_1)$ , so that  $F_1, G_1 \models \phi$ . Let  $\psi$  be the property  $((A_1 \mapsto_{G'_1}^a D_1) * (A_2 \mapsto_{G'_1}^a D_2))$ , so that  $F_1, G'_1 \models \psi$ .

It is clear from the formulation of the example that the action `wall` only affects the edges of  $G_1$ . Thus we obtain an instance of

$$\frac{\phi \quad \text{wall} \quad \psi}{\phi \blacktriangleright \xi \quad \text{wall} \quad \psi \blacktriangleright \xi}.$$

□

## 7. SUMMARY AND FUTURE WORK

We have provided, with examples, a uniform logical framework for reasoning compositionally about both multi-layer and multi-lateral access control policy models. We have shown how our approach accounts for the underlying system architecture, and so provides a way to identify and reason about how vulnerabilities may arise (and be removed) as a result of the architecture of the system.

Further research might address several issues. At the theoretical level, it would be useful to develop the theory to handle the more general case of using, compositionally, the binary layering constructor  $\blacktriangleright$  as a basis for access control models rather than the more restrictive  $n$ -ary stack

construction. Such a development would allow a richer treatment of lattice models than is possible using stacks (cf. the restricted version of UNIX treated in Example 4.10).

It would also be useful to explore the question, also outlined in Section 4.1, of whether can we identify classes of distributed systems and associated access control policies that are uniformly and completely expressible using our **LGL**-based approach? As we have discussed, it would seem to be necessary to develop a more substantially detailed view of class of underlying distributed systems to which the policy models would be intended to apply, formulate an **LGL**-based model of that class of systems, and then formulate a general definition of each class of policy model in that context. As explained in Section 4.1 — we repeat here for convenience — such a programme of work would require the following issues to be addressed:

- First, any such class of distributed systems and associated models would necessarily be somewhat restrictive. Identifying appropriate choices — sufficiently general for an interesting result, yet sufficiently tractable — would already be a significant challenge;
- Second, even given a class of systems of interest, together with a viable class of models, a number of choices concerning the relationship between security features, as described by access control policies, and operational features, which may undermine the intended access control policies, would typically remain. Consequently, further modelling work — in the established sense of observation, model construction, model exploration, and validation — would be required in order to identify appropriate classes of systems and policies to be studied;
- Third, it is unlikely that the simple propositional form of **LGL** presented here will prove sufficiently expressive to be a convenient or even adequate tool for the purpose. It is likely that (at least) an extension with ‘resource modalities’, along the lines of the extension discussed briefly in [21] will be required. In the weak substructural setting of **LGL**, such an extension will require a substantial meta-theoretical development.

From a more applied point of view, we might consider the following questions:

- We might explore in detail the location-based access control models of Gollman [32] — concerned with reference monitors, addressing, and sandboxing in computer memory — using, for example, the various substructural implications to reason about the movement of resources (data) around a system (cf. example in [21]);
- We might seek to incorporate a more substantive account of agents (e.g., as processes, as in [20]), so allowing a treatment of RBAC. A further development along these lines would take us into the world of stateful access control policy models, such as Clark-Wilson [16].
- We might consider a treatment of how the clearance levels associated with agents and data may change; This would allow us to address access control policies such as the ‘low watermark’ in which there is no restriction on read access between levels, but if an agent reads data at lower level, then its clearance level is reduced to that level.

An important question concerns how we might evaluate the usefulness of our approach in supporting the design and implementation of systems and policies for access control. In this respect a key question — perhaps the key question — is how a policy that is specified for a given architecture, as may be specified using the techniques presented here, is implemented or undermined by the users of the system.

Information security can be defined as addressing the problem of ensuring that just the right agents access to just the right information at just the right times. In fact, this definition readily adapts to physical security, which addresses the problem of ensuring that just the right agents (e.g., people) have just the right access to just the right locations at just the right times.

This problem of controlling access to locations and resources, be they physical or logical, arises throughout the networks of complex systems that support the modern world. Organizations try to manage information security risks through policies and security mechanisms, and have discovered over the past decade that these are only effective if their staffs are able and willing to use them correctly. Johnson & Goetz [38] cite Theresa Jones, a security manager at Dow Chemical:

‘My biggest challenge is changing behaviour. If I could change the behaviour of our Dow workforce, then I think I’ve [sic] solved the problem.’

So, access control policies are set by the managers of systems in order to establish their required or desired security régime. However, a specified policy may fail to deliver the intended outcomes for a number of reasons, such as the following:

- The policy may not fit well with the system’s logical or physical architecture: there may be unavoidable preventions or circumventions of intended information flows;
- The policy may not fit well with the necessary or desired working patterns of the organization’s staff, leading to implicit or explicit behavioural patterns that undermine the intended security policy in favour of operational priorities.

It is natural to ask, then, whether models of systems and their security policies can be used as a basis for understanding how the users of systems behave in the context of given security policies. Such a study would be a form of evaluation of the usefulness of the **LGL**-based modelling approach in supporting the design and implementation of systems and policies for access control and might, indeed, form the basis for a system/policy-design methodology that would encourage compliance with security policies whilst not unacceptably compromising operational efficiency.

Research along these lines is the topic of two current projects at UCL. The first, the UK EPSRC/GCHQ-funded project ‘Productive Security’ is exploring how ideas from utility theory can be incorporated into system models in order to model agents’ decision-making as they encounter constraints that are imposed by security policies [14]. For example, an agent may trade-off the benefits and costs of compliance with the policy against the benefits and costs of circumventing the policy in order to achieve greater operational convenience and efficiency. This work will provide a systems modelling methodology that is able to express the value of policies. But policies will, in this setting, not be modelled formally: they will be captured by the model, but not explicitly represented within it. Models are calibrated using data obtained in extensive empirical studies of security behaviour conducted in a wide range of organizations.

The second project, a GCHQ-funded project, ‘Access Control: Models and Compliance’, at UCL, is exploring how to integrate models of policy, models of system architecture, and models of behaviour to deliver an account of policy compliance and circumvention. From a logical perspective, we will need to look at how logics that capture policies can be combined with logics that capture agents’ beliefs in order, overall, to represent properties of systems in which users interact with policies in the context of given system architectures.

These projects form part of a substantial project to bring to bear logical and economic modelling to analyse security design, behaviour, and decision-making. The work presented here is but one small component of that project.

**Acknowledgements.** We are grateful to James Brotherston, Max Kanovich, Edmund Robinson, and Angela Sasse for their comments on this work. We also thank the anonymous referees for their suggestions and questions, which have informed the present form of our presentation.

#### REFERENCES

- [1] M. Abadi. Logic in access control. *Proceedings of the Eighteenth Annual IEEE Symposium on Logic in Computer Science (LICS 2003)*. Phokion G. Kolaitis (editor), IEEE Computer Society Press, 2003.
- [2] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin. A calculus for access control in distributed systems. *ACM TISSEC*, 15(4):706–734, 1993.
- [3] E. Amoroso. *Fundamentals of Computer Security Technology*. Prentice-Hall, 1994.
- [4] R. Anderson. *Security Engineering (Second Edition)*, Wiley, 2008.
- [5] D. E. Bell and L. J. LaPadula. Secure computer systems: Unified exposition and multics interpretation. Technical Report MTR-2997, Mitre Corporation, 1976.
- [6] C. Bertolissi and M. Fernández. Rewrite specifications of access control policies in distributed environments. LNCS 6710:51–67, 2011.
- [7] K.J. Biba. Integrity considerations for secure computer systems. Technical Report MTR-3153, Mitre Corporation, 1977.
- [8] S. Bistarelli, S.N. Foley, and B. O’Sullivan. Detecting and eliminating the cascade vulnerability problem (etc.) In *Innov. Applns. of Artif. Intell.*, 808–813, 2004.
- [9] B. Bollobás. *Modern Graph Theory*. Springer, 1998.
- [10] P. Bonatti, S. De Capitani di Vimercati, and P. Samarati. An algebra for composing access control policies. *ACM TISSEC*, 5(1):1–35, 2002.
- [11] D. F. C. Brewer and M. J. Nash. The Chinese wall security policy. In *Proceedings of the 1989 IEEE Symposium on Security and Privacy*, 206–214, 1989.

- [12] G. Bruns and M. Huth. Access control via Belnap logic: Intuitive, expressive, and analyzable policy composition. *ACM TISSEC*, 14(1):1–27, 2011.
- [13] L. Cardelli, P. Gardner, and G. Ghelli. A spatial logic for querying graphs. LNCS 2380:597–610, 2002.
- [14] T. Caulfield, D. Pym, and J. Williams. Compositional Security Modelling: Structure, Economics, and Behaviour. Proc. Human Aspects of Information Security, Privacy, and Trust Second International Conference, HAS 2014, Held as Part of HCI International 2014, Heraklion, Crete, Greece, June 22–27, 2014. LNCS 8533: 233–245.
- [15] X. Cheng, X. Chen, B. Zhang, and Y. Yang. An algebra for composing access control policies in grid. In *Int. Conf. Comp. Intell. Sec.*, 2009.
- [16] D. D. Clark and D. R. Wilson. A comparison of commercial and military computer security policies. In *Proc IEEE Symp. on Security and Privacy*, 184–195, 1987.
- [17] M. Collinson, B. Monahan, and D. Pym. A logical and computational theory of located resource. *Journal of Logic and Computation*, 19(b):1207–1244, 2009.
- [18] M. Collinson, B. Monahan, and D. Pym. *A Discipline of Mathematical Systems Modelling*. College Publications, 2012.
- [19] M. Collinson and D. Pym. Algebra and logic for resource-based systems modelling. *Mathematical Structures in Computer Science*, 19:959–1027, 2009.
- [20] M. Collinson and D. Pym. Algebra and logic for access control. *Formal Aspects of Computing*, 22(2 (3-4)):83–104 (483–484), 2010.
- [21] M. Collinson, K. McDonald, and D.Pym. A Substructural Logic for Layered Graphs. To appear, *Journal of Logic and Computation*, 2014. doi: 10.1093/logcom/exu002
- [22] J. Crampton, G. Loizou, and G. O’Shea. A logic of access control. *The Computer Journal*, 44(2), 2001.
- [23] A. Dawar, P. Gardner, and G. Ghelli. Expressiveness and complexity of graph logic. *Information and Computation*, 205:236–310, 2007.
- [24] D.E. Denning. A lattice model of secure information flow. *Communications of the ACM*, 19(5):236–243, 1976.
- [25] Reinhard Diestel. *Graph Theory (Third Edition)*. Springer, 2005.
- [26] K. Došen. Sequent systems and groupoid models I. *Studia Logica*, 47:353–385, 1988.
- [27] K. Došen. Sequent systems and groupoid models II. *Studia Logica*, 48:41–65, 1989.
- [28] A. Fiat, D. Foster, H. Karloff, Y. Rabani, Y. Ravid, and S. Vishwanathan. Competitive algorithms for layered graph traversal. *SIAM Journal on Computing*, 28(2):447–462, 1998.
- [29] D. Galmiche, D. Méry, and D. Pym. The Semantics of **BI** and Resource Tableaux. *Mathematical Structures in Computer Science*, 15:1033–1088, 2005.
- [30] F. Giunchiglia, R. Zhang, B. Crispo, and A. Artale. Relation based access control: Logic and policies. Tech. Rep. DISI-10-053, Dept. of Inf. Eng. and Comp. Sci., University of Trento, 2010.
- [31] J. Goguen and J. Meseguer. Security policies and security models. *Proc. of the IEEE Symp. on Security and Privacy*, 11–20, 1982.
- [32] D. Gollman. *Computer Security*. John Wiley & Sons; 2nd Edition edition, 2005.
- [33] D. Harel. Dynamic logic. In D. Gabbay and F. Guenther, editors, *Handbook of Philosophical Logic, Volume II*, pages 497–604. Dordrecht: D. Reidel, 1984.
- [34] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32(1):137–161, 1985.
- [35] J. Hoagland, R. Pandey, and K. Levitt. Security policy specification using a graphical approach. Tech. Rep. CSE-98-3, Comp. Sci. Dept., U. Cal., Davis, 1998.
- [36] J. Horton, R. Harland, E. Ashby, R. Cooper, W. Hyslop, B. Nickerson, W. Stewart, and O. Ward. The cascade vulnerability problem. In *IEEE Comp. Soc. Symp. on Research in Security and Privacy*, 110–116, 1993.
- [37] S.S. Ishtiaq and P. O’Hearn. **BI** as an assertion language for mutable data structures. In *28th ACM-SIGPLAN Symp. Principles of Prog. Langs.*, 14–26, 2001.
- [38] E. Johnson and E. Goetz. Embedding Information Security into the Organization. IEEE Security & Privacy May/June 2007, pages 16–24.
- [39] M. Koch, L.V. Mancini, and F. Parisi-Presicce. Graph-based specification of access control policies. *J. Comp. Sys. Sci.*, 71(1):1–33, 2005.
- [40] A. Korzybski. A Non-Aristotelian System and its Necessity for Rigour in Mathematics and Physics. Presented before the American Mathematical Society at the New Orleans, Louisiana, meeting of the American Association for the Advancement of Science, December 28, 1931. Reprinted in *Science and Sanity*, 1933, 747–761.
- [41] T. Kosiyatrakul, S. Older, and S.-K. Chin. A modal logic for role-based access control. LNCS 3685:179–193, 2005.
- [42] J. Lambek. From categorical grammar to bilinear logic. In P. Schroeder-Heister and K. Došen, editors, *Substructural Logics*, pages 207–238. OUP, 1993.
- [43] B. Lampson, M. Abadi, M. Burrows, and E. Wobber. Authentication in distributed systems: Theory and practice. *ACM Trans. Comp. Sys.* 4(10):265–310, 1992.
- [44] NCSC. Trusted network interpretation of the trusted computer system evaluation criteria. (The Red Book). National Computer Security Center, NCSC-TG-005, 1987.
- [45] P.W. O’Hearn and D.J. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, 1999.
- [46] C. Papadimitriou and M. Yannakakis. Shortest paths without a map. *Theoretical Computing Science*, 84(1):127–150, 1991.

- [47] C. Park, S. Park, and Y. Kim. S-COI : The secure conflicts of interest model for multilevel secure database systems. LNCS 2973:146–153, 2004.
- [48] A. Paz. A theory of decomposition into prime factors of layered interconnection networks. *Discrete Applied Mathematics*, 159(7):628–646, 2011.
- [49] J. Peek, S. Powers, T. O’Reilly, and M. Loukides. *Unix Power Tools, Third Edition*. O’Reilly Media, Inc., 2002.
- [50] D.J. Pym, P.W. O’Hearn, and H. Yang. Possible worlds and resources: The semantics of *BI*. *Theoretical Computer Science*, 315(1):257–305, 2004.
- [51] S. Read. *Relevant Logic*. Basil Blackwell, 1988.
- [52] J. Reynolds. Separation logic: A logic for shared mutable data structures. *Proc. 17th LICS*: 55–74, IEEE Comp. Soc. Press, 2002.
- [53] R. Sandhu. Lattice-based access control models. *IEEE Comp.*, 26(11):9–19, 1993.

MATTHEW COLLINSON, UNIVERSITY OF ABERDEEN  
*E-mail address*: `matthew.collinson@abdn.ac.uk`

KEVIN MCDONALD, UNIVERSITY OF ABERDEEN  
*E-mail address*: `kevin.mcdonald@abdn.ac.uk`

DAVID PYM, UCL  
*E-mail address*: `d.pym@ucl.ac.uk`