Under-Specification, Composition and Emergent Properties

H. M. Hinton
Department of Electrical and Computer Engineering
Ryerson Polytechnic University, Toronto, Canada
hhinton@ee.ryerson.ca

Abstract

Emergent behaviours are those that result from the interaction of the component behaviours within a composite system. We show that emergent behaviours, and their emergent properties, play a role in the composability and satisfiability of the properties of a composite system. Using an emergent properties analysis we can identify which aspects of component behaviour lead to the (undesirable) emergent behaviour for a given composite system. These "undesirable" behaviours are often the result of the under-specification of the behaviour of the system, or assumptions made about the environment in which the system exists. By identifying these under-specifications and assumptions we are able to "strengthen" the specification and implementation of individual systems so that desired properties will be composable.

Keywords: Composition, Emergent Properties, Composable Properties, Under-Specification

1 Introduction

The difficulty with building new, composite systems from known components is a reflection of the difficulty in designing and implementing composable (security) properties. Properties that are satisfied by individual components "mysteriously" fail when such components are interconnected. While looking at the composability of properties, we decided to take one step back and look at the behaviours of interconnected components. This led us to the notion of "emergent" behaviours, which are in turn described by "emergent" properties. We find that emergent behaviours and properties play a large role in the non-composability of properties. In this paper, we describe how and why this is so.

This paper is structured in two main parts. In the next section (the first part of the paper) we define and classify emergent properties according to their general characteristics. This classification will allow us to discuss both composable and emergent properties of composite systems. The sets of composable and emergent properties are not necessarily distinct, as is highlighted by this classification.

In the second part of the paper we motivate the importance of emergent behaviours and properties with two examples: the composition of information flow systems and the evaluation of rules in an access-control implementation. In the first example we examine the role of emergent properties in the context of the composability of the well-known generalised non-interference (GNI) security property [McC87], [McC88]. We find that the conditions allowing for the emergence of GNI on composition can also be used to explain the non-composability of GNI. The second example enhances an existing set of access-control rules to allow for more flexibility in the granting of access permissions. We find that implicit assumptions that were valid in the original system are no longer valid in the enhanced system. This example illustrates how we can use the emergent property analysis to help ensure that we have correctly defined the behaviour of a system and the effects of the interactions of the behaviours of a system. In both cases we must "analyse the effect to understand the underlying cause" [Hop97].

2 Emergent Behaviours and Properties

In this section we introduce and define the notions of *behaviour* and *property* and how they relate to emergent behaviours and properties.

2.1 Behaviours

A component, C, is a stand-alone functional element that is defined by its input and output behaviour [1].

[1] In general, if we refer to a system we mean an interconnection of components. Otherwise, a component is indistinguishable from a system.

The behaviour of a component is represented by the sequences of inputs and outputs at that component. Outputs may be produced in response to inputs or may be "spontaneously" generated by the component [2]. Consider a simple component that increments the input by one and produces the result as the output. The behaviour (B) of this component is easily described: the output is equal to the value of the input plus one:

$$B : o = i + 1$$

Behaviours are often specified as functions of the inputs (i) and outputs (o) of the system.

2.2 Properties

The properties of a component describe the (desired) characteristics of its behaviour and may be specified as functions or relations on the inputs and outputs. When represented as a relation, the property is treated as a predicate, that is, the property can be true or may be false [3].

The following property is a predicate on a function of the increment component (above):

$$P_1 : o = i + 1$$

This property is trivially satisfied by the correct implementation of the component.

For those systems that have more complex behaviours, properties may also be written as predicates on relations, defining the truths common to all behaviours of a component. Such properties are more general than the property P_1, above. For example, the following property of the increment component is more general than the defined behaviour of the component:

$$P_2 : o > i$$

Such a property is too general to specify the exact behaviour of the component but may include the level of detailed required for analysis. There are infinitely many such general properties that can be defined for this component.

It is the responsibility of the component designer to identify those properties that are relevant and contain the required information for the correct use of the component within a larger environment. For example, if we required that the output of the component equal to the input plus two, P_2 does not have enough information to be useful. Either P_1 or a new property,

$$P_3 : o < i + 2$$

give the required information about the behaviour of the component.

[2] When the component is viewed as a black box, producing outputs without any corresponding inputs.

[3] In reality, properties that are specified as a function are also intended to be treated as a predicate, that is evaluated for truth or falsity. In practice, however, properties that are specified as functions should be trivially true as they should correspond directly to the behaviour of the system.

2.3 Under-specification of Behaviours

The increment component, described above, has one and only one possible behaviour. Given an output value, we can uniquely determine the corresponding input value. Such a behaviour is said to be ***fully specified*** as it defines all possible means to arrive at a given output. Equivalently, this behaviour fully specifies all behaviours of the component. A fully specified behaviour corresponds to a total function, that is, for every input (in the set of possible inputs) there is a defined output (in the set of possible outputs).

Often there are several behaviours that may accomplish the same output or goal within a system. In such a system, it may not be possible to determine if the goal was the unintended consequence of yet another (undefined) behaviour. A set of behaviours, leading to the same goal, are ***under-specified*** if when taken together, they do not define all possible inputs that may cause a (set of) outputs.

Under-specification is a powerful tool that allows us to reasonably define the behaviour of a system without have to enumerate every possible input. We may also use under-specification as a means of abstracting away details, or levels of detail, that are not required to correctly model a system. Thus under-specification can be viewed as an artifact of the behaviour ***description*** as opposed to the behaviour itself.

For example, it may be the case that certain (combinations of) inputs will lead to an undesired given output. If these inputs are judged to not be possible, we may choose to under-specify the behaviour of a system by not specifically disallowing these combinations of inputs. We have abstracted out this level of detail from the model of the system and the system's behaviour description.

Under-specification may be the "source" of security problems. Consider the case where previously not possible inputs become possible. If these inputs are not included in the definition of the system behaviour, previously unanticipated behaviour may suddenly appear. As another example, under-specification may result from a failure to consider how a system could be used instead of just how it is intended to be used. Behaviours that were not explicitly excluded (or were included because they were not possible for a component in isolation) may become possible for a component within a composite system. These behaviours must now be explicitly identified so that we can determine if they are harmful and so that we can include their effects in the analysis of any further system composition.

Under-specification of behaviours (and their properties) need not always be a problem. Consider a system where the behaviour of a system in the presence of an alien attack is not specified. Such an under-specification may be considered to be entirely justified and can be safely ignored, given the state of the world in 1997. Under-specification allows us to define behaviours based on what is possible (or reasonably impossible) for a given system.

The reasonableness underlying the under-specification of a system's behaviour may change, however. For example, the reasonable assumption that the we will not be attacked by aliens may become less reasonable after the discovery of intelligent, war-faring life on Mars, or if we require this assumption for a system located at Centauri Prime. What was a reasonable and justifiable under-specification is no longer reasonable nor justifiable: we must revise our opinions of the reasonableness of the under-specification in the system.

2.4 Emergent Behaviours

Emergent behaviours result from the interconnection of components; they are the behaviours relevant to the composite system, not just to the individual system components. Emergent behaviours will always be exhibited by a composite system; emergent properties characterise these emergent behaviours. Emergent behaviours are important because they govern the desired behaviour of a composite system, including the security requirements of a system whose behaviour is more than the sum of its parts.

Emergent behaviours correspond to the behaviours exhibited by the interaction of individual, component behaviours. For example, the logic gates ***and, or, not*** have simple, easily expressed behaviours. The (easily predictable) behaviours and properties of a binary adder emerge when we compose these logic gates to form an adder. The meta-stable behaviour that may be exhibited by a flip-flop (another possible connection of logic gates) is more than the sum of the individual component behaviours. Such a behaviour is not easily predicted from the examination of the component behaviours.

2.5 Emergent Properties

One definition of emergent properties states that "[e]mergent properties are properties of a complex system that emerge or arise only when components are put together and are not visible or identifiable by considering only individual components." [Lev91]. We believe that the definition of emergent properties should include both properties that are previously identifiable and those that are not. We therefore define an emergent property as one that is not satisfied by the behaviour of at least one (and possibly none) of the components of a composite system and yet is satisfied by the composition of these components.

3 Emergence and Composition

Emergent behaviours are easy to determine: they are the behaviours of an interconnected, composite system. What is not as easily determined are the ramifications of these behaviours with respect to the newly interconnected components. For this we must consider the corresponding emergent properties.

We find that there are two types of emergent properties that need to be considered. The first type, known as *type 1 emergent properties*, are those that are relevant to some (but not all) of the components of a composite system and are also relevant to the composite system[4]. A ***type 2 emergent property*** is one that is not relevant to the components of a composite system but is relevant to the composite system.

Throughout this discussion we will use two simple logic gates, AND (Figure 1(b)) and NOT (Figure 1(a)), to illustrate emergent behaviours (B) and properties (P). These components have the following behaviours and properties:

	NOT		AND
B_N	$o = \neg i$	B_A	$o = i_1 \land i_2$
P_{N1}	$o = \neg i$	P_{A1}	$o = i_1 \land i_2$
P_{N2}	$(i = 0) \rightarrow (o = 1)$	P_{A2}	$(i_1 = 1) \land (i_2 = 1) \rightarrow (o = 1)$
P_{N3}	$(i = 1) \rightarrow (o = 0)$	P_{A3}	$(i_1 = 0) \lor (i_2 = 0) \rightarrow (o = 0)$

The NOT gate will invert the input, such that a 0 becomes a 1, and 1 becomes 0 (this logic gate is sometimes known as an inverter or a single-input NOR-gate). The AND-gate will produce a 1 output if and only if both inputs are 1 (input i_1 AND input i_2 are equal to 1); otherwise the output of the AND-gate will be 0.

3.1 Type 1 Emergence

A *type 1 emergent property* is one that is relevant to some, but not all, of a composite system's components and is also relevant to the composite system. How does this happen? Consider the composition of two NOT gates and an AND gate such that the NOT gates are used to invert the inputs to the AND gate, as shown in Figure 1(c). The overall output of these interconnected gates reflects the behaviour of the AND gate and NOT gates, where the values that are AND'ed are the inverted values of the composite system inputs.

The composite system behaviour is easily defined by taking the conjunction of the component behaviours and renaming the inputs and outputs where necessary:

[4] If this property were relevant to all components it would be considered to be a composable property, not an emergent property.

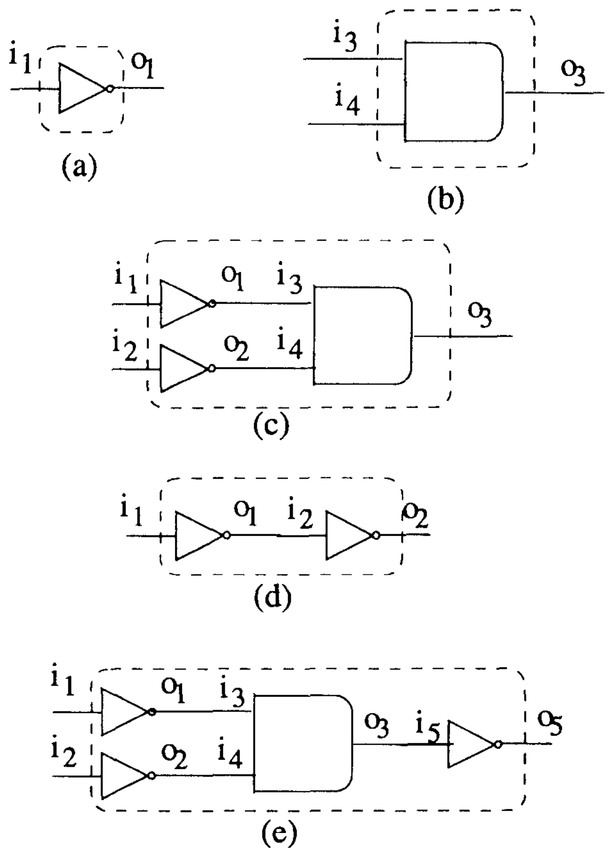

Figure 1: Logic Gates

$$
\begin{aligned}
& B_A(o_3/o, i_3/i_1, i_4/i_2) \land B_N(o_2/o, i_2/i) \\
& \quad \land B_N(o_1/o, i_1/i) \land (i_3 = o_1) \land (i_4 = i_2) \\
\Rightarrow\ & (o_3 = i_3 \land i_4) \land (o_2 = \neg i_2) \land (o_1 = \neg i_1) \\
& \quad \land (i_3 = o_1) \land (i_4 = o_2) \\
\Rightarrow\ & (o_3 = (\neg i_1) \land (\neg i_2))
\end{aligned}
$$

The properties of the AND gate continue to be relevant to the composite system. These properties cannot be classified as composable properties because they are not relevant to the NOT gates. Instead, we classify them as type 1 emergent; they are relevant to the AND gates and the composite system.

Type 1 emergent properties are easy to identify in that they are already defined for the individual components of a system. Once it has been decided that the component properties are relevant to the composite system, we must determine if they are satisfied by the composite system. If not satisfied by the composite system behaviour, these emergent properties must be made to be satisfied, otherwise we cannot allow this composition.

The sets of composable properties and emergent properties are not necessarily distinct. A composable property may act as an emergent property if that composable property is not satisfied by all components of a composite system. Consider the composition of a component that satisfies a known, composable property with a component that does not satisfy this property; this composable property exhibits the characteristics of a type 1 emergent property if the resultant composite system satisfies this property.

3.2 Type 2 Emergence

Consider a property relevant to a composite system but not to any of its components: this is a ***type 2 emergent property***[5]. Determining the satisfiability of previously identified properties [6], while tedious, is simply a matter of determining if the previously identified properties are relevant, and if they are satisfied by the composite system behaviour.

As a simple example of type 2 emergent properties, consider the series interconnection of two NOR gates, shown in Figure 1(d). The behaviour and properties of these two components are identical. Does this mean that the properties are composable? What properties are satisfied by the composite system? If we take the conjunction of the individual component behaviours and employ renaming, we find that

$$
\begin{aligned}
& B_N(o_2/o, i_2/i) \land B_N(o_1/o, i_1/i) \land (i_2 = o_1) \\
\Rightarrow\ & (o_2 = \neg i_2) \land (o_1 = \neg i_1) \land (i_2 = o_1) \\
\Rightarrow\ & (o_2 = \neg(\neg i_1)) \\
\Rightarrow\ & (o_2 = i_1)
\end{aligned}
$$

The behaviour of this interconnection "cancels" the behaviour of the individual components. The properties of these components are not composable. Instead, the not-previously relevant property that the output equals the input emerges for this composite system. This type 2 emergent property follows directly from the fully specified behaviour of the composite system, given the fully specified behaviour of the individual components.

Consider an equivalent example: replace the NOT gates with identically keyed Enigma machines. A naive assumption would be that a plaintext message would be doubly encrypted with the given key. In fact, the two encryptions cancel (as do the NOT gates) so that the result is the original plaintext. Likewise, double encryption with DES using different keys does not produce an encryption that is twice as strong as a single encryption: instead, a much *weaker* encryption emerges [7].

Unfortunately, not all type 2 emergent properties are easy to identify. Consider the slightly more elaborate example shown in Figure 1(e). The overall behaviour of

[5] This corresponds to Leveson's definition of an emergent property.

[6] Identified for similar systems, for previous attempts at composing the same components, or for different composite structures of the same set of components.

[7] These observations are due to Marv Schaefer.

this interconnection is given as

$$o_5 = \neg(\neg i_1 \wedge \neg i_2)$$

This composite system behaviour does not "obviously" reflect the properties of AND or NOT. Is this behaviour instead described by some relevant but previously unknown property? (Such a property is referred to as a *pedantic property*[8]).

How do we identify the pedantic properties of the composite system shown in Figure 1(e)? We begin by comparing the behaviour of the individual components with that of the composite system, to see what new behaviours have "emerged". For this example, a truth table is the simplest means to accomplish this:

Inputs		NOT		AND	Composite	
i_1	i_2	$i_3 = o_1$	$i_4 = o_2$	o_3	o_5	
0	0	1	1	1	0	(1)
0	1	1	0	0	1	(2)
1	0	0	1	0	1	(3)
1	1	0	0	0	1	(4)

From line (4) of the truth table, we see that the composite system has the same behaviour as the AND gate when both inputs to the system have the value 1. This implies that the composite system satisfies an emergent property equivalent to the property of an AND gate:

$$P : o_5 = (i_1 \wedge i_2)$$

This property does not describe all possible sources of a 1 output from the composite system, however. From the truth table, it is also the case that if input i_1 equals 1 OR input i_2 equals 1, then the output o_5 is also equal to 1. This leads to the pedantic property that

$$P : o_5 = (i_1 \vee i_2)$$

Because we have identified this property directly from the system's behaviour, we can be reasonably sure that it is satisfied by the composite system. Of course, if we were familiar with DeMorgan's law, this pedantic property would not come as a surprise. This highlights another point about pedantic properties: what is pedantic to one evaluator need not be pedantic to another.

How do we identify these pedantic properties, in general? There are no good answers to this question. This identification must be based on what we know of similar systems, what is required of this system, and the composite system behaviour. The benefit in the explicit identification of pedantic properties is in furthering our understanding of the system so that we can fully evaluate the system and correctly interconnect it with other systems.

In general, to identify pedantic properties we either need a very simple system or some knowledge of the intended use of and threats to the system. We look for pedantic properties as assurance that a proposed composite system will behave properly and will not misbehave in an identified but unanticipated manner.

If we are not aware of a potential problem then we cannot identify it, nor can we identify which behaviours cause this would violate this problematic behaviour. Therefore, the key to the identification of pedantic properties is the knowledge and understanding of the vulnerabilities of a system and the threats that are introduced through composition. The identification of such vulnerabilities is based on the knowledge of the system's desired behaviour, the behaviour of the system components and the environment of the composite system together with the knowledge of the failures and causes of failures in similar systems[9].

3.3 Assessment of Emergence

The identification of emergent properties is simplified when approached in a straightforward manner. This assessment is easily incorporated into the analysis of the composite system's behaviours and (composable) properties. We begin by defining the behaviour of the composite system. This is defined by the conjunction of the individual component behaviours together with the renaming required given the component interconnections. Given the (previously identified) properties of the individual components, we examine the composable and type 1 emergent properties of the composite system.

The next part of the analysis is to identify the system's type 2 emergent properties. We are trying to identify if there are under-specified behaviours within the composite system. In particular, we must try to identify if there are any unanticipated causes of a given behaviour that may result from the interaction of individual component behaviours.

If, at any stage, there are required properties that cannot be made to be satisfied for the composite system (either by modifying the system or the expectations of the property), there is no need to continue: the composite system cannot meet expectations and should not be used or attempted.

[8] pedant n. 1 a person who insists on strict adherence to formal rules or literal meaning at the expense of the wider view. 2 a person who rates academic learning or technical knowledge above everything. 3 a person who is obsessed by a theory; a doctrinaire (p. 977 The Concise Oxford Dictionary of Current English, Eighth Edition, 1990).

[9] This may require some form of analysis of the potential vulnerabilities and threats to a system. How to accomplish this is beyond the scope of this paper.

3.4 How Does Emergence Affect Composition?

A property is composable if when two components satisfying this property are interconnected, the composite system also satisfies this property. When properties fail on composition, it is because the conditions that allowed these properties are no longer true: in particular, it is most often the case that new behaviours violate these properties. It is not just *any* new behaviour, however, that may violate these properties. We find that these "violating" behaviours are often the result of under-specified component behaviours, and are, in essence, vulnerabilities that have been discovered and exploited. The specific, harmful conditions that were not possible for the components do become possible for the composite system. This leads to emergent behaviours that violate "composable" properties.

4 Emergence and Generalised Non-Interference

In this example, we show how to use the identification of emergent properties to evaluate Generalised Non-Interference (GNI), a possibilistic information flow property. We consider the composite of two systems, A and B, shown in Figure 2. Both systems A and B individually exhibit "functional" GNI-security: any changes to the *hi*-level inputs cannot be detected as such by the *lo*-level users. It is always possible that a *hi*-level output may be added or deleted to nullify these changes.

System A trivially exhibits GNI: there are no *lo*-level events to reveal any *hi*-level information. System B exhibits GNI as a type 1 emergent property, given a reasonable under-specification of B's behaviour. This under-specification allows us to create the composite system of A and B that will violate GNI. The property of $\neg$GNI will emerge for this composite system.

This system is based on Rushby's example of a non-GNI secure composite system [Rus91]. This system is made up of two components: a multi-level system, B, and a single-level system, A. System B produces a *lo*-level output given two *hi*-level inputs by interleaving them (using exclusive-or) with a random, internally-generated, *hi*-level output. System A is used to form a feedback loop over system B. System B can be decomposed into the base components B1, B2, and B3[10]. This example has been used to account for the effects of non-determinism in a composite system, for feedback on composition, and the effects of the delay of events on composition [McC88], [Rus91], [TLB+88], [ZL95].

[10] Note that A cannot be realistically decomposed.

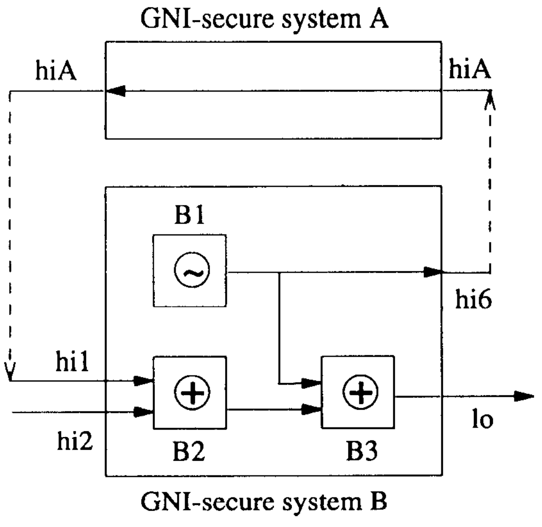

Figure 2: GNI-Insecure Composite System

4.1 GNI as an Emergent Property

In the first part of this example we examine the decomposition of system B, shown in Figure 3. This decomposition is based on the three constituent actions of system B: exclusive-or of two *hi*-level inputs, generation of a random, *hi*-level output, and the exclusive-or *lo*-level output. In this example we take an informal

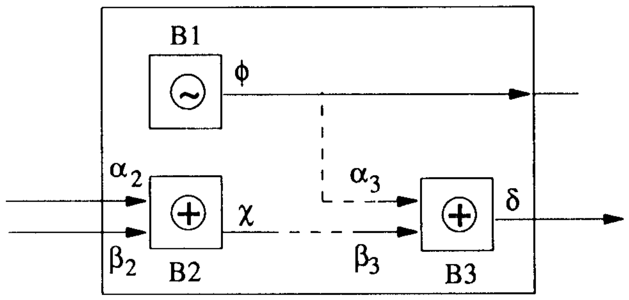

Figure 3: GNI Secure System B

approach to the problem. This approach can be easily applied to a formal specification of the systems and is not limited to a natural language approach.

Component A simply copies its input as its output.

Component B1 (spontaneously) produces a random *hi*-level output.

Component B2 takes two *hi*-level inputs and produces as the output the exclusive-or of the inputs. This behaviour is described by the property that the *hi*-level output is always equal to the exclusive-or of the *hi*-level inputs.

Component B3 takes two *hi*-level inputs and produces as a *lo*-level output the exclusive-or of the inputs. The property defining this behaviour states that the *lo*-level output is the exclusive-or of the *hi*-level inputs.

At this point, we note that, *even though components* B2 *and* B3 *are functionally identical, the output of* B2 *is rated hi, while the output of* B3 *is rated lo*. For component B3 we have the property that the output, the exclusive-or of two *hi*-level events, is a *lo*-level output. B3 as described is *NOT* GNI-secure; any change in the *hi*-level inputs is immediately reflected in the *lo*-level outputs. Components B1 and B2 are trivially GNI-secure because a *lo*-level cannot determine anything about the *hi*-level inputs or outputs of either component.

Composite System B

Given the interconnections specified in Figure 2 the behaviour of the system is defined by the *hi*-level random output and the *lo*-level output which is in turn equal to the exclusive-or of the *hi*-level inputs and the *hi*-level output.

Emergent Property Analysis

Examining composite system B, we find that

- there are no known composable properties common to all components;
- there are no properties common to all components that may be relevant to the composite system;
- there are no properties of B1 or B2 that are relevant to the composite system;
- the property describing B3's output is relevant to the composite system (because the composite system produces a *lo*-level output, as does B3);
- GNI, satisfied by B1 and B2, is relevant to the composite system.

Does GNI act as an emergent property for the composite system? According to our "functional" implementation of GNI at component B1 and B2, we must determine if the *lo*-level output of B implies anything about the *hi*-level events of B. We want to show that $\neg(\delta \rightarrow \alpha_2)$, that the *lo*-level output does not imply anything about the *hi*-level input α_2. By the properties of exclusive-or, we know that $\alpha_2 \dot{\vee} \beta_2 \dot{\vee} \phi$ does not imply anything about α_2 or β_2 or ϕ individually.

Thus GNI is a type 1 emergent property for this composite system: it is satisfied by some of the components (B1 and B2) but not all (B3) and is relevant to and satisfied by the composite system. The type 1 emergence of GNI is due to the "independence" of the *hi*-level events within B.

Is there any under-specification involved with this system? It turns out that the desired behaviour and the satisfiability of GNI both rely on a reasonable under-specification of the inputs to the system. We do not explicitly require that no two of the *hi*-level sequences are identical: this is an under-specification of the requirements on the system's environment. We postulate that any future failure of GNI-security will take advantage of this vulnerability.

4.2 GNI as a Non-Composable Property

In this section we examine the composition of the two individually GNI-secure systems A and B. Because GNI is satisfied for both A and B, it cannot act as a type 1 emergent property. GNI may be composable or not composable: if GNI is not composable, then $\neg$GNI appears as a type 2 emergent property for the proposed composite system.

If we connect the *hi*-level output of B with the input of A and the output of A with one of the *hi*-level inputs of B, we create the composite system of Figure 2. The (emergent) behaviour of the composite system states that the *lo*-level output is still the exclusive-or of the *hi*-level events, but now the *lo*-level output is identical to one of the *hi*-level inputs. This violates our implicit requirement that the *hi*- and *lo*-level events be independent by taking advantage of the under-specification of B.

Not only is GNI not composable for the proposed composite system, but the property that the *lo*-level output reflects the *hi*-level input follows directly from the fully-specified behaviour of the composite system. Thus if we were to fully specify component B we would add the requirement that no two of the *hi*-level sequences be the same. This requirement is no different from the one identified in the previous section. If we had included this requirement with the specification of B we would have immediately seen that the proposed composite system was not allowable. By not including this requirement we have had to resort to the emergent property analysis to determine the properties of the composite system AB.

It has been argued that introducing a delay element into component A will have the effect of allowing the composite system to be GNI-secure, but previous approaches have been unable to determine how much delay is required [ZL95]. This emergent analysis tells us *why* a delay will allow for GNI-composability: it partially restores the independence of the *hi*-level sequences, removing the vulnerability associated with this under-specification.

5 Emergence and Access-Control

Consider as another example of emergent behaviour the granting and revocation of access permissions within a system. This example does not involve the interconnection of physical components. Instead, we compose behaviours in the guise of enhancing the functionality of a system. We see that type 2 emergent behaviours result from the under-specification of the initial system behaviours when new behaviours governing the granting and revocation of access-rights are added to the system. Again, we limit our exposure to formal specification and analyses in favour of a more intuitive explanation of the concepts and issues involved.

5.1 Initial System

Our initial system is quite simple: a subjects may *own* objects. The owner of an object in turn may *grant* or *revoke* access permissions on that object to both itself and other subjects. A subject that has been granted access *has* that permission. Subjects are identified as s_1, s_2, objects are o_1, o_2, and access rights a may include *r*ead, *w*rite, and e*x*ecute, where the set of access rights is given as $A = \{r,w,x\}$.

A subject, s_1, that *owns* an object, o, may *grant* to another subject, s_2, access rights on o:

$$own(s_1, o) \rightarrow grant(s_1, o, a, s_2)$$

Similarly, if s_1 owns o and s_2 *has* access right a on o, then s_1 may *revoke* this right:

$$own(s_1, o) \wedge has(s_2, a, o) \rightarrow revoke(s_1, o, a, s_2)$$

These behaviours can be described by properties describing the changes to a subject's access rights. For example, after s_1 has granted a on o to s_2, then s_2 *has* this permission:

$$own(s_1, o) \wedge grant(s_1, o, a, s_2) \rightarrow has(s_2, a, o)$$

If s_2 had a on o, then it no longer has this right after s_1 has revoked it:

$$own(s_1, o) \wedge has(s_2, a, o) \wedge revoke(s_1, o, a, s_2) \rightarrow \neg has(s_2, a, o)$$

In this simple system, only the owner of an object may grant or revoke permissions on that object. What happens when we decide to expand the functionality of this system?

5.2 Proposed Enhancements

Let us add to this system the functionality that will allow subject s_1 to grant *partial ownership* of s_1's objects to another subject, s_2. A partial owner of an object ($p\_own(s_1, o)$) will be allowed to grant and revoke access rights to the owner's object to other subjects:

$$own(s_1, o) \rightarrow grant\_p\_own(s_1, o, s_2)$$
$$own(s_1, o) \wedge p\_own(s_2, o) \rightarrow revoke\_p\_own(s_1, o, s_2)$$

Just as the owner can grant and revoke access rights, so can the partial owner:

$$p\_own(s_1, o) \wedge has(s_2, a, o) \rightarrow revoke(s_1, o, a, s_2)$$
$$p\_own(s_1, o) \rightarrow grant(s_1, o, a, s_2)$$

The properties of this enhanced functionality are similar to those above, with the precondition of ownership ($own(s, o)$) replaced by the condition of partial ownership ($p_own(s, o)$). In addition, there are equivalent properties on the granting and revocation of partial ownership.

Emergent Property Analysis

Given these behaviours and properties, what new behaviours and properties will result from adding the partial ownership functionality to the system? We consider the conjunction of these behaviours and find that

- there are no known "composable" properties;
- there are no properties common to both "systems" that may be relevant to the composite system (no non-composable properties that must be satisfied by the composite system);
- the behaviours and properties of the initial system are relevant and required of the composite system (identifying potential type 1 emergent properties);
- the behaviours and properties of the enhanced functionality are relevant and required of the composite system (identifying potential type 1 emergent properties);

These last two points identify potential type 1 emergent properties of the composite system. We begin our assessment of this system by attempting to ensure that the properties of the original system and the enhanced functionality are satisfied when combined.

By the conjunction of the granting behaviours, we find that either owner or partial owner may grant access rights on an object:

$$[(own(s_1, o) \vee p\_own(s_1, o)] \rightarrow grant(s_1, o, a, s_2)$$

The composite behaviour satisfies granting-behaviour properties for both ownership and partial ownership, so that the related type 1 emergent properties are satisfied for the enhanced system.

Are there any type 2 emergent properties associated with granting, that is, is this behaviour fully satisfied? It does define all possible means by which a subject may be granted access to an object (by definition). Are there any unintended consequences of this composite behaviour? It turns out that this composite behaviour specifies more than we may have originally intended.

Consider s_1 as the owner of an object. s_1 may grant access rights on that object to ***any other subject***, s_2, including subjects that partially own that object. Consider s_1 as the partial owner of an object. s_1 may grant access rights on that object to ***any other subject***, s_2, including subjects that *own* that object. These behaviours correspond to type 2 emergent behaviours and are described by type 2 emergent properties. It was not explicitly intended with either the original system or the enhanced functionality that a partial owner of an object have such sweeping powers. These type 2 emergent properties imply that the behaviour defining the granting of access rights is under-specified. To be fully specified, we cannot allow a partial owner to override an owner's access rights to an object.

Similarly, the behaviour of partial owner revocation is also under-specified. If s_2 partially owns o, s_2 may revoke the owner's access rights on o. While we may have been able to rationalize the emergence of the partial owner's granting abilities, it is hard to imagine when we would allow this type of partial owner revocation.

We may decide that we do not want a partial owner to be able to grant or revoke rights on an object to that object's owner. If so, we must modify the enhanced functionality to remove the under-specification that leads to this undesirable emergent behaviour. [11]

We limit the partial owner's ($s_1's$) abilities by not allowing them to grant or revoke access rights to a subject s_2 if s_2 is the owner of that object:

$$p\_own(s_1, o) \wedge \neg own(s_2, o) \rightarrow grant(s_1, o, a, s_2)$$
$$p\_own(s_1, o) \wedge \neg own(s_2, o) \wedge has(s_2, o) \rightarrow revoke(s_1, o, a, s_2)$$

Unlike the previous example, there are no over-arching properties, such as GNI, that are violated by the composite system. Thus it may not be as clear that these emergent behaviours are not desirable for the enhanced access-control system.

The behaviour of this composite system is still underspecified in a potentially unacceptable manner. Consider the following, subtle, scenario: s_1 owns o, s_2 partially owns o, s_1 grants a on o to s_3, which s_2 is then able to revoke. That is, the continued under-specification of the composite behaviour allows a partial owner to revoke access rights granted by any other subject that has ownership or partial ownership on the same object. The removal of this under-specification is accomplished in the same manner as demonstrated above.

In this example above we have seen how the (under)specification and composition of behaviours can lead to the (type 2) emergence of undesirable behaviour, satisfying undesirable emergent properties. In the initial system, it is reasonable to define revocation rights in terms of ownership. Ownership is the only way that we can change access-rights: why build in fail-stops for other, not-possible, types of ownership? It is only when the conditions for grant and revocation of rights change that this under-specification becomes dangerous, as evidenced by the emergent property analysis.

6 Discussion

There has been a great deal of work on the composability of properties, in particular properties that deal with "information flow" ([McC87], [McC88], [JT88], [WJ90], [Rus91], [ZL95]). The main drawback with this work is that is has focussed on one particular property (for example, Generalized Non-Interference) or one class of properties (possibilistic properties).

McLean has proposed a more general approach to composition based on *Selective Interleaving Functions* (SIFs) to determine the composability of properties [McL94]. This approach relies on the trace-based specification of properties and has focussed on possibilistic properties. As part of this work McLean has also classified the "strength" of different properties and has demonstrated that when a weaker property is composed with a stronger property, the weaker property is preserved. This is equivalent to stating that the weaker property acts as a type 1 emergent property for the composite system.

Abadi and Lamport have proposed a more general approach to composition [AL90], using the Temporal Logic of Actions [Lam90]. This approach takes "a more general semantic view in which a specification is a set of behaviours." (p 10, [AL90]). It is from this notion that we have developed the idea that even though a specification may be a set of behaviours, if the behaviours are not fully specified, then neither is the specification.

In their syntactically independent approach, a behaviour is represented by an infinite sequence of states, where a state is an assignment of values to variables. These values are in turn drawn from *sets* of possible values. Using sets of possible values lends itself intuitively to our discussion of under-specification where we often differentiate between states (and behaviours) that may occur and those we would rather did not occur.

7 Conclusions

In this paper we have introduced the notion of the under-specification and emergence of behaviours and properties. This approach can be used in conjunction with existing approaches to composition, such as McLean's Selective Interleaving Functions or the Abadi-Lamport Composition Principle. The advantage this approach is that not only can we evaluate the composability of properties, we can identify the causes of non-composability and evaluate the efficacy of proposed "fixes."

The assessment of emergent properties is based on an understanding of the security requirements of a composite system. Emergent behaviours are those that describe the behaviour of a composite system. These behaviours are characterized by emergent properties. The proposed emergent property analysis can be used to ensure that a properties composite system meets its specified requirements. A thorough examination of emergent properties offers insights into desired and undesired composite system behaviour, including the under-specification of behaviours that lead to emergent behaviours [12].

[11] Note that we may wish to allow the *owner* to revoke access rights that have been granted by a *partial owner*. This too is an emergent behaviour, however it fits in with our intended operation of the system and should be made explicit.

[12] Note that we have not explicitly considered what happens when we are connecting systems with conflicting policies, as dis-

The role of an emergent property analysis is to identify which aspects of component behaviour lead to the undesirable emergent behaviour for a given composite system. Undesirable emergent behaviours often result from the under-specification of possible behaviours, so that new, undesirable behaviours become possible for a composite system. This emergent behaviour approach allows us to determine the causes of the lack of composability of (emergent) properties. This will in turn allow us to predict and prevent undesirable emergent behaviours and properties. This information can be used to design the safeguards necessary to protect a system's vulnerabilities from mis-use[13].

We used this emergent property analysis to explain the type 1 emergence of GNI in a simple system. We then went on to show how to explain the non-composability of GNI based on the conditions of its type 1 emergence. This in turn allows us to evaluate any proposed "fixes" that are intended to ensure the composability of GNI. We also used the emergent property analysis to study the proposed enhancements to an existing access-control system. In this second example, we found that the behaviours of the initial system are under-specified in the context of the composite system. This in turn allowed type 2 emergent behaviours which violated the intended operation of the enhanced system. The good news with this example is that the emergent property analysis also allowed us to identify how to strengthen the behaviours of the initial system to be fully-specified within the composite system, removing the undesired emergent behaviours.

Some of the difficulties in identifying the desired properties of a composite system arise because the undesired system behaviours are only identified once the system is in use. This is a failing in the design of the system – we cannot hope to build a secure system if we cannot define how it is to behave or not to behave. Nevertheless, we believe that many goals can be identified by the clear definition of the desired goals of the system, knowledge of the properties of the components of the system and knowledge of the composable and emergent properties of similar systems.

Acknowledgments

This paper has benefited greatly from the insightful comments of Judy Hemenway, John Hopkinson and the anonymous referees. Also, Bob Blakely, John Dobson, and Marv Schaefer provided valuable feedback on the ideas in this paper during the workshop.

cussed by Hosmer [Hos92].

[13] Related work in the flow of "insecurities" through a system appears in "An Insecurity Flow Model" by Moskowitz and Kang, in these proceedings [MK97].

References

[AL90] Martín Abadi and Leslie Lamport. Composing specifications. Technical Report 66, Digital Systems Research Center, Palo Alto, California, October 1990.

[Hop97] John Hopkinson. Personal communication. E-mail of 23 July 1997, 1997.

[Hos92] Hilary Hosmer. The multipolicy paradigm for trusted systems. In *ACM SIGSAC New Security Paradigms Workshop*. IEEE Computer Society Press, 1992.

[JT88] Dale M. Johnson and E. Javier Thayer. Security and the composition of machines. In *Proceedings of the Computer Security Foundations Workshop*, pages 72–89. IEEE Computer Society Press, June 1988.

[Lam90] Leslie Lamport. A temporal logic of actions. Technical Report 57, Digital Systems Research Center, Palo Alto, California, April 1990.

[Lev91] Nancy G. Leveson. Issues in assuring safety in composite trustworthy systems. 1992 NATO Workshop on Composite Trustworthy Systems, October 1991.

[McC87] Daryl McCullough. Specifications for multilevel security and a hook-up property. In *Proceedings of the 1987 IEEE Symposium on Security and Privacy*, pages 161–166. IEEE Computer Society Press, May 1987.

[McC88] Daryl McCullough. Noninterference and the composability of security properties. In *Proceedings of the 1988 IEEE Symposium on Security and Privacy*, pages 177–186. IEEE Computer Society Press, May 1988.

[McL94] John McLean. A general theory of composition for trace sets closed under selective interleaving functions. In *Proceedings of 1994 Computer Society Symposium on Research in Security and Privacy.* IEEE Computer Society, IEEE Computer Society Press, 1994.

[MK97] Ira Moskowitz and Myong Kang. An insecurity flow model. In *Proc. 1997 New Security Paradigms Workshop*, September 1997.

[Rus91] John Rushby. Composing trustworthy systems: A position paper. In *Proceedings of the NATO RSG2 Working Conference on Composability*, October 1991.

[TLB$^+$88] B. Thomson, E.S. Lee, P.I.P. Boulton, M. Stumm, and D.M. Lewis. A trusted network architecture. Technical report, Computer Systems Research Insititute University of Toronto, Toronto, Ontario, Canada, October 1988.

[WJ90] J. Todd Wittbold and Dale M. Johnson. Information flow in nondeterministic systems. In *Proceedings of the 1990 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 144–161. IEEE Computer Society Press, May 1990.

[ZL95] Aris Zakinthinos and E. Stewart Lee. The composability of non-interference. In *8th IEEE Computer Security Foundations Workshop*, 1995.