Remarks on the GWV Firewall

Ron van der Meyden
School of Computer Science and Engineering,
University of New South Wales

October 18, 2010

Abstract

Greve, Wilding and Vanfleet (2003) have proposed a formal security policy for separation kernels, and used it prove a security claim for a firewall implemented in a system satisfying this security policy. The paper revisits their example and formulates a more general version of the result, that simplifies its structure and clarifies its information theoretic content.

1 Introduction

Separation kernels [Rus81] are a kind of minimalist operating system, intended to provide the abstraction of a set of partitions within which computation can proceed independently, without interference from other partitions, and with communication between these partitions forced to conform to a given security policy. The classical notion of separation assumed the simplest possible policy for interprocess communication – complete prohibition – but this has long been understood to be too restrictive for practical purposes. More recent work has dealt with policies that permit some communication between processes, but constrain this to specific channels [GWV03, MWTG00].

Given the key role that separation kernels play as infrastructural components in systems security, it has been of interest to subject them to formal verification. This raises the question of what specific properties a kernel should be formally shown to posses – one that is best answered by asking how such properties enable further reasoning about systems constructed on the basis of a separation kernel. Surprisingly little of the literature seems to have addressed this question.

One example is the work by Greve, Wilding and Vanfleet (henceforth GWV) [GWV03] who have defined a security policy that has served as the basis for their kernel verification efforts. GWV have presented as evidence for the appropriateness of their policy model an application to a particular class of configurations, intended to represent systems with a firewall through which all information from a secure domain to an insecure domain must flow.

In this paper, we revisit this example, and argue that its formulation can be significantly simplified and clarified. Our presentation of the example makes it clear that the example makes information theoretic assumptions, a fact that was hidden in the GWV presentation, where a key property is stated as a safety property plus the assumption that a function satisfying certain constraints exists. We argue that our presentation is simpler and more general.

We describe the GWV policy in Section 2 and present the firewall example in Section 3. Our improved presentation is treated in Section 4. Some conclusions are drawn in Section 5.

2 The GWV Policy Model

The GWV separation policy has undergone a number of revisions: a discussion of the variants can be found in [Gre10]. We base our treatment here on the original version [GWV03], in which the firewall example is cast, and which suffices for the points we wish to make.

The GWV system model assumes that a system is decomposed into a set $\mathtt{Part}$ of partitions, and that each partition $p \in \mathtt{Part}$ is associated with a set of *segments*, denoted $\mathtt{segs}(p)$. We write $\mathtt{Seg}$ for the set of all segments. Partitions may overlap, i.e., there may be segments that are associated to more than one partition. The set of states of a machine is denoted S. In each state $s \in S$, each segment $a \in \mathtt{Seg}$ has a value, which we denote $\mathtt{val}_a(s)$. System evolution is deterministic: given a state, the following state after one step of computation is represented by a function $\mathtt{next} : S \to S$. Each step of computation is viewed as being executed within or by a particular partition, represented by a function $\mathtt{cur} : S \to \mathtt{Part}$. That is, given a machine state s, the currently active partition is $\mathtt{cur}(s)$, and the result of one step of computation by this partition is the state $\mathtt{next}(s)$. We represent a machine conforming to this system model by the tuple $M = \langle \mathtt{Part}, \mathtt{Seg}, \mathtt{segs}, S, \mathtt{cur}, \mathtt{val}, \mathtt{next} \rangle$.

Definition of the security policy for such a machine begins by describing how information is permitted to flow from segment to segment. Formally, this is captured by a relation $\rightarrowtail \subseteq \mathtt{Seg} \times \mathtt{Seg}$. Intuitively, $a \rightarrowtail b$ means that information from segment a is permitted to flow to segment b. Given a set X of segments and two states $s, t \in S$, we say that *s and t are equivalent on X*, and write $s \equiv_X t$, if $\mathtt{val}_a(s) = \mathtt{val}_a(t)$ for all $a \in X$. The GWV separation policy is now defined by the following condition.

> **Sep:** For all segments a and states $s, t \in S$, if $\mathtt{cur}(s) = \mathtt{cur}(t)$, and s and t are equivalent on $\{a\} \cup \{b \in \mathtt{segs}(\mathtt{cur}(s)) \mid b \rightarrowtail a\}$, then $\mathtt{val}_a(\mathtt{next}(s)) = \mathtt{val}_a(\mathtt{next}(t))$.

To understand this property, it is helpful to define a more general notion that will also prove useful in the sequel. Suppose that X_p is a set of segments for each partition p. (If $X_p = X$ for all p we represent this parameterized set simply by X.) If a is a segment, say that a machine M satisfies the condition that *the next value of a depends only on X and the current partition* if for all states s, t, if $\mathtt{cur}(s) = \mathtt{cur}(t)$ and s and t are equivalent on $X_{\mathtt{cur}}(s)$ then $\mathtt{val}_a(\mathtt{next}(s)) = \mathtt{val}_a(\mathtt{next}(t))$. Intuitively, this states that the next value of the value of a is a function of the current partition p and the current values of segments in X_p. We may now rephrase **Sep** as stating that the next value of a segment a depends only on $X_p = \{a\} \cup \{b \in \mathtt{segs}(p) \mid b \rightarrowtail a\}$ and the current partition p. That is, segments on which the next value of a is allowed to depend are a itself, together with segments b satisfying $b \rightarrowtail a$ (i.e., from which the policy permits flow of information to a) that are also associated with the currently active partition.[1]

3 GWV and Rushby's Firewall Example

By way of justification of their framework, GWV consider a firewall-like example. They show that if a machine satisfies the separation policy with respect to a particular flow policy $\rightarrowtail$, plus a condition concerning the correct operation of a partition representing the firewall, then information is released in a secure fashion. Their work was conducted in the theorem prover ACL2, and has been reproduced by Rushby [Rus04] using the theorem prover PVS. Both formulations of the example make use of certain auxiliary functions and require a nontrivial set of additional assumptions about these functions. The impact of these assumptions on the class of systems to which the result applies is unclear. In this section we describe this example and presentations of it by GWV and Rushby. We offer an improved presentation in the following section.

The partition to which information is released by the firewall is called B, for "Black". The firewall itself is represented by a partition F. A nonspecified set of other partitions may exist in the system. Information from the firewall to the black partition is constrained to flow through a particular segment `outbox`. This is formally represented by the firewall policy condition

> **FW-Pol:** For all segments $a \in \mathtt{segs}(B)$, segments b and partitions $P \neq B$, if $b \rightarrowtail a$ and $b \in segs(P)$, then $a = \mathtt{outbox}$ and $P = F$.

That is, the only partition besides B from which information is permitted to flow to B is F, and such flow is only permitted via the segment `outbox`.

GWV then consider a distinction between "black" and "red" data. "Black" data, intuitively, is the data that is permitted to be released to the Black par-

[1] Some subtleties about this definition are worth noting. We could have distinct partitions p, p' such that $\mathtt{segs}(p) = \mathtt{segs}(p')$. Depending on which partition is active, the next value of the segment a could differ (consider two partitions running two different programs over the same shared data). Thus, expressing this notion as "the next value of a depends only on X" would not quite capture the content of the definition.

Note also that in case $b \rightarrowtail a$ for no b in the current partition, the definition states that the next value of a depends only on $\{a\}$. This does allow that the value of a changes, but any changes must be deterministic. Ticking of a clock, or progress in a deterministic computation using only the segment, would be examples of this. GWV state that it also allows "asynchronous arrival of (partition specific) information from external sources." Given the determinism of the function `next`, this would seem to require the very strong assumption that the future arrival of such information is already encoded into the segment using, e.g., prophecy variables.

tition. Safe operation of the firewall is then represented by the property that if all data in the Black partition is black, then it will always remain black.

Suppose we formally represent the fact that the data in a segment a is black in a state s by the notation $s \models \texttt{black}(a)$. Then we we may capture the statement that the firewall maintains the invariant that all data in the segment `outbox` is black by the property

> **FW-Blackens:** For all states s, if $\texttt{cur}(s) = F$ and $s \models \texttt{black}(\texttt{outbox})$ then $\texttt{next}(s) \models \texttt{black}(\texttt{outbox})$.

GWV now claim that if a machine satisfies **Sep**, **FW-Pol** and **FW-Blackens**, then it maintains the invariant that all data in the Black partition are black. More formally,

> **FW-Correct:** For all states s, if $s \models \texttt{black}(a)$ for all $a \in \texttt{segs}(B)$, then $\texttt{next}(s) \models \texttt{black}(a)$ for all $a \in \texttt{segs}(B)$.

To prove this claim, they require an auxiliary function `scrub` mapping a segment and a state to a state. We write the resulting state as $\texttt{scrub}_a(s)$ for a segment a and state s. The intuition for this function is that it blackens the segment: a concrete interpretation they discuss involves zeroizing the segment in order to remove sensitive information. GWV state the following properties for this function.

> **S1:** `(scrub-commutative)` For all states s and segments a, b, we have $\texttt{scrub}_a(\texttt{scrub}_b(s)) = \texttt{scrub}_b(\texttt{scrub}_a(s))$

> **S2:** `(segment-scrub-different)` For all states s and segments a, b, if $a \neq b$ then $\texttt{val}_a(\texttt{scrub}_b(s)) = \texttt{val}_a(s)$.

> **S3:** `(black-scrub)` For all states s and segments a, b, $\texttt{scrub}_b(s) \models \texttt{black}(a)$ iff $a = b$ or $s \models \texttt{black}(a)$.

> **S4:** `(current-scrub)` For all states s, we have $\texttt{cur}(s) = \texttt{cur}(\texttt{scrub}_a(s))$.

> **S5:** `(spontaneous-generation)` For all states s, if $s \models \texttt{black}(a)$ for all segments a, then $\texttt{next}(s) \models \texttt{black}(a)$ for all segments a.

> **S6:**`(black-function-of-segment)` For all states s, t and segments a, if $\texttt{val}_a(s) = \texttt{val}_a(t)$, then $s \models \texttt{black}(a)$ iff $t \models \texttt{black}(a)$.

Rushby [Rus04] (apparently drawing on information in GWV's published ACL2 proof scripts [GWV03]) gives a formulation that uses a function $\texttt{blacken} : S \rightarrow S$. Intuitively, `blacken` turns all segments black. Its required properties are:

> **B1:** For all $a \in \mathit{Segs}$, and all states s, we have $\texttt{blacken}(s) \models \texttt{black}(a)$.

B2: For all segments a and states s, if $s \models \texttt{black}(a)$ then $\texttt{val}_a(s) = \texttt{val}_a(\texttt{blacken}(s))$.

B3: For all states s, we have $\texttt{cur}(s) = \texttt{cur}(\texttt{blacken}(s))$.

B4: For all states s, t and segments a, if $\texttt{val}_a(s) = \texttt{val}_a(t)$, then $s \models \texttt{black}(a)$ iff $t \models \texttt{black}(a)$.

B5: For all states s, if $s \models \texttt{black}(a)$ for all segments a, then $\texttt{next}(s) \models \texttt{black}(a)$ for all segments a.

The relationship between these formulations can be seen in the following result.

Proposition 1 *Suppose that the set of segments in a machine is finite and that the machine satisfies* **S1**-**S6**. *Define the function* `blacken` *to map a state to the state resulting from scrubbing all non-black segments, i.e., if the set of non-black segments in state s is $\{a_1, \ldots, a_n\}$ then*

$$\texttt{blacken}(s) = \texttt{scrub}_{a_1}(\texttt{scrub}_{a_2}(\ldots \texttt{scrub}_{a_n}(s))).$$

Then the machine satisfies **B1**-**B5**.

Proof: B1 follows from **S3**. **B2** follows from **S2**, noting that `blacken` does not scrub already black segments. **B3** is immediate from **S4**. **B4** is identical to **S6**. **B5** is identical to **S5**. □

Thus, Rushby's assumptions are weaker than GWV's (under the reasonable assumption of finiteness of the set of segments, which necessarily holds in GWV's theory as a result of representing states using finite lists). The proof of the GWV result now proceeds by showing that **Sep**, **FW-Pol** and **FW-Blackens** imply **FW-Correct**, in the presence of **B1-B5**. Both papers note that the use of the additional functions and properties raises a question: is the extended set of axioms even consistent? If not, the reason that these automated proofs are successful may be simply the inconsistency of the assumptions. To allay this concern it is shown that there exists a model for the collection of assumptions.

However, we note that even this check does not suffice to establish the original claim. It may well be that the assumptions **B1-B5** can be satisfied, but at the cost of restricting the class of systems to which the result applies. What is really required, if the function `blacken` satisfying **B1-B5** is to be just a device of the proof, is that *every* system satisfying **Sep**, **FW-Pol** and **FW-Blackens** can, by suitable definition of `blacken`, be extended to one also satisfying **B1-B5**. This has not been proved by GWV or Rushby. Indeed, inasmuch as the existence of the function `blacken` satisfying **B1** implies the existence of a state in which every segment is black, it is apparent that the extra assumptions are in fact restrictive on the class of machines to which the result applies. We return to this issue below, where we give a sharper statement of this observation.

4 An Alternative Formulation of the Firewall

We propose here and prove correct an alternative formulation of the GWV correctness claim. Our approach is to eliminate use of auxiliary functions like `scrub` and `blacken` altogether, and replace the conditions **B1**-**B5** by a property somewhat like **B5**, but which captures more of the intuition underlying the predicate `black`.

Condition **B5** can be understood as saying that once all data in the system are black, they remain black forever. More general than this is the following: if some data are derived solely from black data, then they are black. We may capture this formally using the notion of dependency introduced above. (Note here we quantify over sets X that do not depend on the partition p, and use the convention introduced above that $X_p = X$ for all p.)

> **Black:** For all sets $X \subseteq \texttt{Seg}$, segments a and states s, if the next value of a depends only on X and the current partition, and if $s \models \texttt{black}(b)$ for all $b \in X$, then $\texttt{next}(s) \models \texttt{black}(a)$.

We would argue that this condition is entirely compatible with GWV's intuitions. Note that it implies **B5**. It is also apparent that this assumption places no constraints on the machine. Note that for every machine, there exists an interpretation of the predicate $s \models \texttt{black}(a)$ such that **Black** is satisfied, viz., take $s \models \texttt{black}(a)$ to hold for all states s and segments a. The following shows that the condition **Black** is weaker than **B1**-**B5**.

Proposition 2 *If a machine satisfies* **B1**-**B5** *then it satisfies* **Black**.

Proof: Assume **B1**-**B5**. Suppose that the next value of a segment a depends only on X and the current partition. Suppose that s is a state such that $s \models \texttt{black}(b)$ for all segments $b \in X$. We need to prove that $\texttt{next}(s) \models \texttt{black}(a)$.

For this, consider the state $\texttt{blacken}(s)$. Since all segments in X are black in s, by **B2**, s and $\texttt{blacken}(s)$ are equivalent on X. By **B3**, we also have that have that $\texttt{cur}(s) = \texttt{cur}(\texttt{blacken}(s))$. Thus, since the next value of a depends only on X and the current partition, it follows that $\texttt{val}_a(\texttt{next}(s)) = \texttt{val}_a(\texttt{next}(\texttt{blacken}(s)))$.

Now by **B1**, all segments are black in $\texttt{blacken}(s)$. By **B5**, it follows that $\texttt{next}(\texttt{blacken}(s)) \models \texttt{black}(a)$. By **B4** and the fact that $\texttt{val}_a(\texttt{next}(s)) = \texttt{val}_a(\texttt{next}(\texttt{blacken}(s)))$, we now obtain that $\texttt{next}(s) \models \texttt{black}(a)$. $\Box$

The following result shows that the single condition **Black** can play the role of all of **B1**-**B5** in GWV's result.

Theorem 1 *If a machine satisfies* **Sep**, **FW-Pol**, **FW-Blackens** *and* **Black**, *then it satisfies* **FW-Correct**.

Proof: Suppose that the system satisfies **Sep**, **FW-Pol**, **FW-Blackens** and **Black**. Let s be a state such that $s \models \texttt{black}(c)$ for all $c \in \texttt{segs}(B)$. We show that $\texttt{next}(s) \models \texttt{black}(a)$ for all $a \in \texttt{segs}(B)$. We consider four cases for the segment $a \in \texttt{segs}(B)$ and the current partition $\texttt{cur}(s)$, showing that the conclusion holds in each case.

Case 1: Assume $a \in \texttt{segs}(B) \setminus \{\texttt{outbox}\}$. Let $X = B$. By **FW-Pol**, we see that if $b \rightarrowtail a$ then $b \in \texttt{segs}(B)$. It follows using **Sep** that the next value of a depends only on X and the current partition. It is now immediate using **Black**, $X = B$ and the assumption that $s \models \texttt{black}(c)$ for all $c \in \texttt{segs}(B)$ that $\texttt{next}(s) \models \texttt{black}(a)$.

For the remaining cases, we assume $a = \texttt{outbox} \in \texttt{segs}(B)$ and consider the different possible values for $\texttt{cur}(s)$: B, F, or neither.

Case 2: Assume $a = \texttt{outbox} \in \texttt{segs}(B)$ and $\texttt{cur}(s) = B$. Let $X = \{a\} \cup \{b \in \texttt{segs}(\texttt{cur}(s)) \mid b \rightarrowtail a\}$. By **Sep**, the next value of a depends only on X and the current partition. Since $\texttt{cur}(s) = B$, we have $X \subseteq \texttt{segs}(B)$. By the assumption that $s \models \texttt{black}(c)$ for all $c \in \texttt{segs}(B)$, we have $s \models \texttt{black}(c)$ for all $c \in X$. It now follows from **Black** that $\texttt{next}(s) \models \texttt{black}(a)$.

Case 3: Assume $a = \texttt{outbox} \in \texttt{segs}(B)$ and $\texttt{cur}(s) = F$. Since $\texttt{outbox} \in \texttt{segs}(B)$, we have by assumption that $s \models \texttt{black}(\texttt{outbox})$. It is now immediate from **FW-Blackens** that $\texttt{next}(s) \models \texttt{black}(\texttt{outbox})$.

Case 4: Assume $a = \texttt{outbox} \in \texttt{segs}(B)$ and $\texttt{cur}(s) \notin \{F, B\}$. Let $X = \{a\} \cup \{b \in \texttt{segs}(\texttt{cur}(s)) \mid b \rightarrowtail a\}$. Then by **Sep**, the next value of a depends only on X and the current partition. By **FW-Pol**, we have $X = \{a\} = \{\texttt{outbox}\}$. Since $\texttt{outbox} \in \texttt{segs}(B)\}$, it follows from the assumption that $s \models \texttt{black}(c)$ for all $c \in X$. It now follows from **Black** that $\texttt{next}(s) \models \texttt{black}(a)$. $\square$

What we have shown so far is that the condition **Black**, which is weaker than both the GWV and the Rushby assumptions, suffices to prove the firewall correctness claim. One might still ask whether this condition is strictly weaker: conceivably, if **Black** holds in a machine then it can be extended to one satisfying **B1-B5**.

We now show that this is not the case under the most straightforward interpretation of the term "extend": the simple addition of a function $\texttt{blacken} : S \to S$ to the machine, without transforming any of the components of the machine. Note first that whereas **B4** requires that the predicate $\texttt{black}$ be a function of the state and segment, no such requirement is imposed by the condition **Black**. A concrete setting in which this requirement is in fact undesirable is when we wish $\texttt{black}$ to express some past-time property of a segment. A machine $M = \langle \texttt{Part}, \texttt{Seg}, \texttt{segs}, S, \texttt{cur}, \texttt{val}, \texttt{next} \rangle$ can be *unfolded* using a standard type of construction, producing a machine $M^* = \langle \texttt{Part}, \texttt{Seg}, \texttt{segs}, S^*, \texttt{cur}^*, \texttt{val}^*, \texttt{next}^* \rangle$ with the same set of partitions, segments and same relationship of association, but with the other components defined as follows:

1. S^* is the set of finite sequences $s_0, s_1, \ldots, s_n$ such that each $s_i \in S$ and $s_{i+1} = \texttt{next}(s_i)$;
2. $\texttt{cur}^*(s_0, s_1, \ldots, s_n) = \texttt{cur}(s_n)$;
3. $\texttt{val}^*_a(s_0, s_1, \ldots, s_n) = \texttt{val}_a(s_n)$;

X	$f(a)$	$f(b)$	$f(c)$	$\texttt{next}(f)(a)$	$g(a)$	$g(b)$	$g(c)$	$\texttt{next}(g)(a)$
$\{a,b\}$	1	0	0	0	1	0	1	1
$\{a,c\}$	1	0	0	0	1	1	0	1
$\{b,c\}$	0	0	1	0	1	0	1	1

Figure 1: The next value of a depends on at least $\{a, b, c\}$

4. $\texttt{next}^*(s_0, s_1, \ldots, s_n) = s_0, s_1, \ldots, s_n, \texttt{next}(s_n)$.

Intuitively, M^*, behaves exactly like M, but maintains a record of past states. This enables us to define past time properties as predicates on the state. Consider a machine M in which the segment values are natural numbers and the only operation that can be performed is to add the values of two segments and store the result in a third. In this case we can define $s \models \texttt{black}(x)$ to be the predicate "the current value of segment x is greater than or equal to the initial value of segment a" which is a function of the segment x and the state $s_0, s_1, \ldots, s_n$ in M^*, and satisfies **Black**, but is not a function of the current segment value of x in that machine.

A second obstacle to justifying the function `blacken` as an artifact of proof by straightforward extension of the machine is that there exist machines satisfying **Black** in which no state has all segments black (and hence fail to provide the state required to satisfy **B1**). Consider a machine with three segments $\texttt{Seg} = \{a, b, c\}$ and a single partition B. Let the set of states S be the set of functions $f : \texttt{Seg} \to \{0, 1\}$ that are not everywhere 1, and for such a state f define $\texttt{val}_f(x) = f(x)$ for $x \in \texttt{Seg}$. Define the function `next` by $\texttt{next}(f)(a) = f(a) \wedge (f(b) \otimes f(c))$ and $\texttt{next}(f)(b) = f(b) \wedge (f(a) \otimes f(c))$ and $\texttt{next}(f)(c) = f(c) \wedge (f(a) \otimes f(b))$ where '$\otimes$' is exclusive-or. (Note that if we had $\texttt{next}(f)(a) = \texttt{next}(f)(b) = \texttt{next}(f)(c) = 1$ we would have $f(a) = f(b) = f(c) = 1$ as well as $f(b) \otimes f(c) = 1$. This is plainly a contradiction. Hence `next` does map S to S.) For all states f we let $\texttt{cur}(f) = B$. Let the predicate `black` be defined by $f \models \texttt{black}(x)$ iff $f(x) = 1$. Since there does not exist a state f with $f(a) = f(b) = f(c) = 1$, the function `blacken` cannot be defined.

We claim that this machine satisfies **Black**. To see this, we note that for all segments x, the smallest set X of segments such that the next value of x depends only on X and the current value is `Seg` itself. This is demonstrated for the segment a in Table 1, by provision for each maximal set X contained in `Seg` a pair of states f, g such that f and g are equivalent on X but such that $\texttt{val}_a(\texttt{next}(f)) \neq \texttt{val}_b(\texttt{next}(g))$. For the other segments the claim follows by symmetry. Thus the only set X that we need to consider in the antecedent of **Black** is $X = \texttt{Seg}$, for which the condition stating that all segments in X are black in the state s is false by construction. Hence **Black** holds trivially in this machine. We note that by defining the policy so that $x \rightarrowtail y$ for all segments x, y, the machine also satisfies **Sep**, **FW-Pol** and **FW-Blackens**.

It might still be argued that "extension of a machine by a function `blacken`" should be defined in such as way as to allow addition of new states, and that it may be possible to prove the equivalence of **Black** and **B1-B5** for such a notion of extension. For this to be meaningful in the present context, the notion of extension should support a proof of the firewall theorem in the original machine. Given that **Sep** and **Black** both rely on the notion of dependence, which is very sensitive to the set of states in the machine, finding such a notion of extension would appear to be a nontrivial matter.

5 Conclusion

The example studied in this paper can be understood as a case study in compositionality in reasoning about systems using causal notions. The formulation of the firewall example in Section 4 makes it apparent that the predicate $s \models \texttt{black}(a)$, like the separation policy **Sep**, is inherently about the causal history of segments. Thus, our alternate formulation highlights that the example is about what can be derived when one composes two specifications that concern dependency properties in a system.

It would be interesting to have a more general theory about reasoning using such properties. Although the idea of separation kernels has existed since the 1980's and compositionality in reasoning about security has received considerable attention [McC90, McL96, Mil90, WJ90], there seems to be surprisingly little literature that deals directly with what formal properties can be derived about systems that have been built on the foundation of systems satisfying separation properties. Investigation of further examples and a closer comparison with known principles of compositionality in reasoning about security would appear to be a necessary next step in the development of a more general understanding of this area.

There has been a considerable amount of research on process algebraic definitions of noninterference in the past ten years [FG01, Rya01], but very little of this considers separability directly. Roscoe and Wulf [RW95] do present some results showing that certain types of noninterference policies entail separability in the sense of Jacob [Jac90]. However, this notion of separability states that the system is equivalent to an interleaved parallel composition of separated processes, in which each process is enabled at each moment of time. This means that these results cannot be applied directly to systems operating subject to a scheduler, as we have considered here.

References

[FG01] R. Focardi and R. Gorrieri. Classification of security properties (Part I: information flow). In R. Focardi and R. Gorrieri, editors, *Foundations of Security Analysis and Design, FOSAD 2000, Bertinoro, Italy, September 2000*, volume 2171 of *LNCS*, pages 331–396. Springer, 2001.

[Gre10] D. Greve. Information security modelling and analysis. In David S. Hardin, editor, *Design and Verification of Microprocessor Systems for High-Assurance Applications*, pages 249–299. Springer, 2010.

[GWV03] D. Greve, M. Wilding, and M. Vanfleet. A separation kernel formal security policy. In *Proc. Fourth International Workshop on the ACL2 Theorem Prover and Its Applications*, 2003. Paper and associated proof scripts at `http://www.cs.utexas.edu/users/moore/acl2/workshop-2003/`.

[Jac90] J. Jacob. Separability and the detection of hidden channels. *Information Processing Letters*, 34:27–29, Feb 1990.

[McC90] D. McCullough. A hookup theorem for multilevel security. *IEEE Trans. Software Eng.*, 16(6):563–568, 1990.

[McL96] J. McLean. A general theory of composition for a class of "possibilistic" properties. *IEEE Trans. Software Eng.*, 22(1):53–67, 1996.

[Mil90] J.K. Millen. Hookup security for synchronous machines. In *Proc. IEEE Computer Security Foundations Workshop*, pages 84–90, 1990.

[MWTG00] W. Martin, P. White, F.S. Taylor, and A. Goldberg. Formal construction of the mathematically analyzed separation kernel. In IEEE Computer Society Press, editor, *Proc. 15th IEEE Conf. on Automated Software Engineering*, 2000.

[Rus81] J. Rushby. Design and verification of secure systems. In *Proc. 8th Symposium on Operating Systems Principles*, pages 12–21, Asilomar CA, Dec 1981. (ACM Operating Systems Review, Vol 15, No. 1).

[Rus04] J. Rushby. A separation kernel formal security policy in PVS. CSL Technical Note, SRI International, March 2004.

[RW95] A.W. Roscoe and L. Wulf. Composing and decomposing systems under security properties. In *Proc. IEEE Computer Security Foundations Workshop*, pages 9–15, 1995.

[Rya01] P.Y. Ryan. Mathematical models of computer security. In R. Focardi and R. Gorrieri, editors, *Foundations of Security Analysis and Design, FOSAD 2000, Bertinoro, Italy, September 2000*, volume 2171 of *LNCS*, pages 1–62. Springer, 2001.

[WJ90] J. T. Wittbold and D. M. Johnson. Information flow in nondeterministic systems. In *IEEE Symposium on Security and Privacy*, pages 144–161, 1990.