Formal Methods for MILS: Formalisations of the GWV Firewall

Ruud Koolen
Eindhoven University of Technology
r.p.j.koolen@tue.nl

Julien Schmaltz
Eindhoven University of Technology
j.schmaltz@tue.nl

ABSTRACT
To achieve security certification according to the highest levels of assurance, formal models and proofs of security properties are required. In the MILS context, this includes formalisation of key components – such as separation kernels – and the formalisation of applications built on top of these verified components. In this paper, we use the Isabelle/HOL proof assistant to formalise the Firewall application built on top of a verified separation kernel according to the model of Greve, Wilding, and Vanfleet (GWV). This Firewall application has been formalised twice after the original effort by GWV. These different efforts have been compared and discussed on paper. Our main contribution is to provide a formal comparison between these formalisations in the formal logic of a proof assistant.

1. INTRODUCTION
To achieve security certification at the highest levels of assurance (e.g. EAL6 or EAL7 of the Common Criteria), formal models and proofs are required. In the context of MILS architectures, this not only means the formalisation of key components, like separation kernels, but also the formalisation of more mundane applications and their composition in a complete system.

Within the EURO-MILS project, we aim at providing a modelling and validation environment based on the formalisation of a generic MILS architecture. This environment should ease the development of formal models and proofs of systems built according to the MILS architectural paradigm. We present and discuss three existing efforts about the formal verification of an application built on top of a verified separation kernel. This application is a Firewall originally proposed by Greve, Wilding, and Vanfleet [5], who also proved its correctness using the ACL2 theorem prover [6]. This formalisation was later replicated by Rushby [9], who proved the relevant properties in the logic of the PVS [8] proof system; to do so, he also refined the axiomatisation of the Firewall behaviour. Subsequently, a further refinement of this formalisation was proposed by Van der Meyden [2], who also proves relations between the three efforts using informal pen-and-paper proofs. Our main contribution is to formalise Van der Meyden’s axiomatisation and proofs in the Isabelle/HOL proof assistant [7]. We also re-formulate in Isabelle/HOL the formalisations by GWV and Rushby and the relations between the three axiomatisations. As part of this effort we found a small flaw in Van der Meyden’s axiomatisation, for which we present a corrected version; we regard this as a confirmation of the value of the formalisation of mathematics in the logic of computer proof systems.

In the next three sections, we introduce the Firewall example application, the GWV model of formalised security, and express the Firewall in term of the GWV model. Afterwards, we compare the three different axiomatisations of the Firewall, proving relevant relations between them. We point out a flaw in the axiomatisation of Van der Meyden, and present a corrected version. Finally, we formally show how all three axiomatisations are sufficient to prove the desired properties of the larger system containing the Firewall, which we take as the compositional overall correctness proof.

2. THE FIREWALL APPLICATION
The application Greve, Wilding, and Vanfleet used as an example of their formalisation of security is one that sanitises useful but sensitive information for use by an untrusted application. This so-called Firewall takes as input information presented by trusted parts of the system, which may be sensitive. It then filters and censors this information to produce a version that can safely be passed to applications that are not trusted to handle it securely, and delivers this sanitised information to a location where the untrusted application can find it. Under the assumption that the Firewall application is the only source of information to the untrusted application, this should provably ensure that no sensitive information ever ends up within reach of the untrusted application.

The Firewall application does not exist in a vacuum. It runs on top of an operating system of some sort, specified in more detail in the next section, which provides controlled access to memory. Its job is to divide the system memory into segments and enforce limits on which applications can access which memory segments. To ensure security, it is assumed that the operating system is configured in such a way that the Firewall application is the only component that...
can write to memory segments that are accessible to the untrusted application; moreover, it can only write to a single such a segment, denoted as outbox. This configuration is depicted in Figure 1.

For the purpose of the Firewall example, Greve, Wilding and Vanfleet do not try to express in detail what information is and is not sensitive. Instead, they assume the presence of a predicate black that, for a given memory segment and system state, expresses whether or not that segment contains only nonsensitive information. Thus, in a given state, a memory segment is black if and only if its contents are not sensitive. The function of the Firewall, then, is to make sure it only ever writes black data to outbox. The security requirement we seek to formalise can then be expressed as requiring that none of the memory segments accessible to the untrusted application ever becomes nonblack. The main goal of the different formalisations presented in the remainder of this paper is to prove that this is the case under the assumption that the operating system and Firewall work as specified.

3. THE GWV MODEL OF SEPARATION

The system model proposed by Greve, Wilding and Vanfleet [5] (GWV) guarantees a security property called Separation. Extensions and variations of this model have then been proposed [10, 4] and discussed [3, 1]. We will nonetheless use the original GWV model [5], which is sufficient for the purposes of this paper.

The GWV model defines a mathematical formulation of systems similar to the one presented in Figure 1. At its base, a GWV system is a deterministic state machine, with states denoted s or t. Execution consists of repeatedly changing from a state s to the next state denoted next(s). A GWV system contains a finite set of memory segments, which in each state s have contents denoted select(s, a) for segment a. Furthermore, it contains a finite set of partitions, which represent independent subcomponents akin to processes in general-purpose operating systems. In any state, a single partition is currently active and executing; this partition is denoted current(s) for state s. This basic model is formalised in Isabelle parlance using the following axioms:

\[ \text{fixes current :: State \Rightarrow Partition} \]
\[ \text{fixes select :: State \Rightarrow Segment::finite \Rightarrow Value} \]

\[ \text{fixes next :: State \Rightarrow State} \]

Here, State, Partition, Segment, and Value represent arbitrary sets. The finiteness condition on Partition is irrelevant for the correctness of any of the formalised proofs and has been omitted. Finiteness of Segment, on the other hand, turns out to be crucial.

The GWV model assumes that the different partitions run on top of a separation kernel, a basic operating system tasked with the duty of restricting memory access of partitions to those accesses that satisfy a given security policy. One part of this security policy is a set of memory segments \( \text{segs}(p) \) for each partition \( p \) describing the memory segments that that partition is allowed to access. A more subtle component of the security policy is an information flow policy, represented by a binary relation between memory segments, with the semantics that any computation step that writes to memory segment \( a \) may only do so while reading from a limited set of input memory segments. Thus, information may only flow along the edges of the directed graph represented by the information flow policy. GWV formalise the information flow policy as a function \( \text{dia} \), short for direct interaction allowed, which for each memory segment \( a \) returns a set \( \text{dia}(a) \) of memory segments that are allowed to directly influence it:

\[ \text{fixes segs :: Partition \Rightarrow P(Segment)} \]
\[ \text{fixes dia :: Segment \Rightarrow P(Segment)} \]

The security requirement enforced by the separation kernel can then be expressed as requiring that all memory accesses must respect the security policy. Greve, Wilding, and Vanfleet call this property separation. Rather than describing how a separation kernel might enforce such a security policy, GWV define separation as requiring that all actions performed by a partition must be independent of the contents of memory segments that are not allowed to influence the action. Specifically, they require that whenever a partition writes to a memory segment \( a \), the contents written may depend only on the the contents of the segments that are both in the accessible segments of the executing partition and \( \text{dia}(a) \):

\[ \text{assumes Separation :: State, s \in State, a \in Segment.} \]
\[ \text{current(s) = current(t) \land} \]
\[ \text{select(s, a) = select(t, a) \land} \]
\[ \forall b \in \text{dia}(a) \cap \text{segs(current(s))), select(s, b) = select(t, b) \rightarrow select(next(s), a) = select(next(t), a)} \]

This definition reads that for any segment \( a \), for any states for which both the contents of \( a \) and the active partition are equal, the contents of \( a \) in the next state must be a function of the contents of the memory segments that are both readable by the active partition and allowed to influence \( a \). Thus, in changing the contents of \( a \), the executing partition may not make use of any information other than that allowed by the security policy.

In the GWV system model, the presence and correct functioning of a separation kernel is taken as an assumption, as formalised by the Separation axiom. Greve, Wilding, and Vanfleet propose that this axiom is a useful base for proving security properties of larger systems that rely on a separation kernel as a key component. They use their Firewall
application as an example of how to prove security properties of a larger system by relying on the separation kernel as a provider of the base security infrastructure.

4. FIREWALL IN GWV
In this section, we formally define both the Firewall application and the security property it is supposed to provably establish.

The Firewall application is a partition \( F \) that collects sensitive information from unspecified locations in the system, and passes a sanitised version of this information along to a different partition, \( B \), that cannot be trusted to handle it securely. Relying on the separation kernel to ensure that no other partitions can write to memory segments accessible to the untrusted application, this should ensure that the untrusted application can only ever get access to information that has been judged safe by the Firewall.

GWV satisfy this information flow property as the specific requirement that there is a single memory segment \( \text{outbox} \) accessible to \( B \) which may be influenced by segments accessible to partitions other than \( B \). Furthermore, any such segments that can influence \( \text{outbox} \) can only be accessible by \( F \) and \( B \). Formally:

\[
\begin{align*}
\text{assumes FW_Pol: } & \forall a, b \in \text{Segment}. \ P \in \text{Partition}. \\
& a \in \text{segs}(B) \land \\
& b \in \text{dia}(a) \land \\
& b \in \text{segs}(F) \land \\
& P \neq B \rightarrow \\
& (P = F \land a = \text{outbox})
\end{align*}
\]

Together with the Separation axiom described in the previous section, this should suffice to ensure that the only information that ends up in segments accessible to \( B \) is information that the Firewall put there.

As described in Section 2, the behaviour of the Firewall is modelled using the \( \text{black} \) predicate, which models the distinction between sensitive and nonsensitive information. A memory segment is black in a given state if the contents of that segment in that state does not contain any sensitive information. The security functionality of the Firewall, then, is that it never writes any information to \( \text{outbox} \) that would cause it to become nonblack. This can be formalised as the proposition that \( \text{outbox} \) never changes from black to nonblack while the Firewall partition \( F \) is executing:

\[
\begin{align*}
\text{assumes FW_Blackens: } & \forall s \in \text{State}, a \in \text{Segment}. \\
& \text{current}(s) = F \land \text{black}(s, \text{outbox}) \rightarrow \\
& \text{black}(\text{next}(s), \text{outbox})
\end{align*}
\]

We can now state a formal definition of the correctness of the Firewall application. The desired security property of the complete system including the Firewall, the untrusted application, and any possible other applications is that the segments accessible to \( B \) never become nonblack. The requirement that the segments of \( B \) start black is not a property of the Firewall; the weaker property that can be guaranteed by the Firewall is that the segments of \( B \) are already black, they will remain black. Introducing a function \( \text{run} \) to express the execution of a number of computation steps, this can be formalised as follows:

\[
\begin{align*}
\text{fun run :: } N & \Rightarrow \text{State} \Rightarrow \text{State} \text{ where} \\
& \text{run}(0, s) = s \\
& \text{run}(\text{Suc}(n), s) = \text{run}(n, \text{next}(s))
\end{align*}
\]

\[
\text{theorem FW_Correct: } \forall s \in \text{State}, n \in N, a \in \text{segs}(B). \\
\text{black}(s, a) \rightarrow \text{black}(\text{run}(s, n), a)
\]

The combination of the Separation, FW_Blackens, and FW_Pol axioms is insufficient to prove the FW_Correct property. For this to be the case, we need further properties describing the behaviour of the \( \text{black} \) predicate; for example, if black data in the segments accessible to \( B \) could become nonblack on its own accord, the FW_Correct security property quickly falls apart. It is in the axiomatisation of the \( \text{black} \) predicate that GWV, Rushby, and Van der Meyden differ in their approaches. These three approaches will be the topic of the next section.

5. AXIOMATISATING BLACKNESS
The behaviour of the Firewall is defined in terms of the \( \text{black} \) predicate, which models the property of a segment of memory of not containing any sensitive information. It would be expected for this property to satisfy certain axioms, such as the proposition that a segment cannot change from black to nonblack without the segment contents being modified. Certainly, if a nonsensitive chunk of memory were suddenly to become sensitive without anyone touching that piece of memory, this would violate our assumptions on what sensitivity is supposed to mean.

In their publications, GWV, Rushby, and Van der Meyden take different approaches in characterising the assumed behaviour of the \( \text{black} \) predicate. The basic notion of all three approaches is that nonblack data cannot be generated from black data; any computational process that takes only nonsensitive data as its input must surely produce output that is also nonsensitive. In this section, we compare the three different axiomatisations of this notion, and prove relevant relations between them.

5.1 The GWV Formalisation
One of the main properties that GWV require the \( \text{black} \) predicate to have is that in a system in which all segments are black, all segments will remain black; this is a special case of the “no spontaneous generation of nonblack data” principle described above. Another property they require is that blackness is a function of the content of a memory segment; it is not allowed that the same data is considered black or nonblack depending on the context, as this would allow sensitive data to leak into a completely idle partition.

\[
\begin{align*}
\text{assumes S5: } & \forall a \in \text{Segment}. \text{black}(s, a) \rightarrow \\
& \forall a \in \text{Segment}. \text{black}(\text{next}(s), a)
\end{align*}
\]

\[
\begin{align*}
\text{assumes S6: } & \text{select}(s, a) = \text{select}(t, a) \\
& \text{black}(s, a) = \text{black}(t, a)
\end{align*}
\]

These two properties are not sufficient to prove all desired properties of blackness, however. In particular, we would like to be able to prove a version of S5 restricted to a particular partition: the proposition that when all segments of a partition \( P \) are black in a state in which \( P \) is active, then
all these segments will still be black in the next state. A
lemma like this has an obvious role to play in any potential
proof of FW_Correct.

To make this possible, GWV assume that for every state
s and any segment a, a state can be constructed in which a is
black but which is otherwise identical to s; such a state could
be constructed by, say, wiping the contents of the memory
segment a. To formalise this notion, they posit the existence
of a function scrub producing such a state scrub(a, s), with
straightforward properties:

\textbf{fixes scrub :: Segment ⇒ State ⇒ State}
\textbf{assumes S1:}
\textbf{scrub(a, scrub(b, a)) = scrub(b, scrub(a, s))}
\textbf{assumes S2:}
\(a \neq b \rightarrow \text{select}(\text{scrub}(b, s), a) = \text{select}(s, a)\)
\textbf{assumes S3:}
\textbf{black(\text{scrub}(b, s), a) \iff (a = b \lor \text{black}(s, a))}
\textbf{assumes S4:}
\textbf{current(\text{scrub}(a, s)) = current(s)}

Axioms S1, S2, and S4 together specify that scrub does
not change anything relevant about a state other than the
contents of the scrubbed segment; axiom S3 specifies that a
scrubbed segment is black.

With these properties, the lemma sketched above can be
proven. For if all segments accessible to partition \(P\) are
black in a state \(s\) with \(\text{current}(s) = P\), we can construct a
state \(t\) in which all segments are black by scrubbing all other
segments; per axiom S5, the next state \(\text{next}(t)\) of \(t\) also has
all \(P\)-accessible states black. But \(s\) and \(t\) have the same
contents of all memory segments accessible to \(P\); according
to the Separation axiom, the same must hold for \(\text{next}(s)\)
and \(\text{next}(t)\). Thus, per S6, because all \(P\)-accessible segments in
\(\text{next}(t)\) are black, the same must hold for \(\text{next}(s)\).

The axioms S1 ... S6 together with the FW_Blackens,
FW_Pol, and Separation properties are sufficient to prove the
desired FW_Correct theorem. We shall prove this by
showing that the GWV axioms are stronger than the Rushby
axioms, and that the Rushby axioms are sufficient to prove
FW_Correct, both claims of which are described in the
section below.

\textbf{5.2 Rushby's Version}

The formalisation proposed by Rushby is a reasonably
minor refinement of the original by GWV. Like GWV, Rushby
includes the two main properties from the GWV formalisa-
tion:

\textbf{assumes B4:}
\textbf{select(s, a) = select(t, a) \rightarrow 
black(s, a) = black(t, a)}
\textbf{assumes B5:}
\((\forall a \in \text{Segment}. \text{black}(s, a)) \rightarrow 
(\forall a \in \text{Segment}. \text{black}(\text{next}(s), a))\)

The difference between the two formalisations is that instead
of a function scrub that replaces the contents of a single
segment by a blackened version, Rushby posits a function
blacken that for a given state scrubs all segments that are
not black.

\textbf{fixes blacken :: State ⇒ State}
\textbf{assumes B1:}
\textbf{black(blacken(s), a)}
\textbf{assumes B2:}
\textbf{black(s, a) \rightarrow select(s, a) = select(blacken(s), a)}
\textbf{assumes B3:}
\textbf{current(s) = current(blacken(s))}

It is clear that this axiomatisation is quite similar to GWV’s
original. The lemma that was proven for the GWV formal-
isation can be proven for Rushby’s version in a very similar
way. Furthermore, it is clear that the Rushby axioms fol-
low from the GWV axioms: the blacken function can be
constructed by calling scrub on each segment that is not
black\(^1\), and the resulting function clearly satisfies the B1,
B2, and B3 axioms.

Unfortunately, formalising this fact in Isabelle proved chal-
lenging. In the logic of Isabelle, the fact that Segment is
finite is expressed as a nonconstructive assertion that a func-
tion \(f :: \mathbb{N} \Rightarrow \text{Segment}\) and a natural number \(n\) exist such
that the segments \(f(m)\) for \(0 \leq m < n\) exactly cover Seg-
ment. Because this is a nonconstructive assertion, we can
only similarly prove in a nonconstructive way that a function
blacken must exist that behaves like a repeated application of
scrub. Both proving that this function exists, and work-
ging with this function to prove properties such as B1 ... B3
about it, are technically challenging; moreover, they result
in cumbersome proofs that are difficult to understand due
to the technical tricks intermixing the substance of the arg-
ument. We consider this a weakness of the Isabelle proof
system; a more readable way of dealing with nonconstruc-
tively existing entities would be a welcome improvement.
Moreover, to avoid turning this section into an unreadable
mess of proof trickery, we have omitted the formal proof that
the GWV axioms imply the Rushby axioms.

Similarly to the previous section, we prove that the axioms
B1 ... B5 combined with the FW_Blackens, FW_Pol, and
Separation properties are sufficient to prove the desired
FW_Correct theorem by reducing this problem to the re-
lated problem for Van der Meyden’s axiom. Doing so will
be the subject of the remainder of this paper.

\textbf{5.3 Van der Meyden's Axiom}

Van der Meyden argues that these axioms defined by GWV
and Rushby unnecessarily restrict the class of systems to
which the results apply. He proposes an alternative formu-
lisation consisting of one axiom over predicate black without
the need of ancillary functions, and thus the richness of the
state space they imply. Rushby’s axiom B5 basically states
that if the entire system is black, then this is still the case
in the next state. Van der Meyden generalises this notion
by requiring that if the value of a memory segment is com-
pared based only on the values of a set of memory segments
\(X\), and all segments in \(X\) are black, then the computed
segment must be black in the next state.

The notion of \textit{being computed based only on a set of memory
segments} is just the same that the Separation axiom. In the Separation axiom, the requirement of

\(^1\)Here the fact that the number of segments if finite is crit-
ical, without this requirement, the function blacken could
not be defined based on scrub.
The security policy is that the content of a memory segment in the next state is a function of its current content, the active partition, and the contents of the memory segments that are allowed to influence it. Using the same construction, Van der Meyden defines his sole axiom formalising \texttt{black} as follows:

\[
\begin{aligned}
\text{fun equals} & : \mathbb{P} (\text{Segment}) \Rightarrow \text{State} \Rightarrow \text{State} \Rightarrow \mathbb{B} \\
\text{where} & \\
\text{equals} (X, s, t) & = \forall a \in X. \text{select} (s, a) = \text{select} (t, a)
\end{aligned}
\]

\textbf{assumes Black:} $\forall X \in \mathbb{P} (\text{Segment}), s \in \text{State}, a \in \text{Segment}$,

\[
(\forall r, t \in \text{Segment} \cdot \text{equals} (X, r, t) \land \\
\text{current} (r) = \text{current} (t) \rightarrow \\
\text{select} (\text{next} (r), a) = \text{select} (\text{next} (t), a)) \\
\text{assumes} (\forall b \in X. \text{black} (s, b)) \\
\text{black} (\text{next} (s), a)
\]

This axiom states that if the value of segment $a$ in the next state of $s$ is a function of the segments in $X$ and the active partition, then this function preserves blackness. In other words, as the value of $a$ is computed based on $X$, so is the blackness of $a$ inherited from $X$.

The \textbf{Black} axiom follows from the Rushby axioms; indeed, the proof for this in Isabelle is very simple. The proof starts by assuming the two premises stated in the \textbf{Black} axiom:

\[
\begin{aligned}
\text{fix} & \ X \ s \ a \\
\text{assume} & 1: \\
(\forall r, t \in \text{Segment} \cdot \text{equals} (X, r, t) \land \\
\text{current} (r) = \text{current} (t) \rightarrow \\
\text{select} (\text{next} (r), a) = \text{select} (\text{next} (t), a)) \\
\text{assume} & (\forall b \in X. \text{black} (s, b)) \\
\text{black} (\text{next} (s), a)
\end{aligned}
\]

Because $a$ is already black in $s$ and is thus unaffected by \texttt{blacken} due to B2, and because \texttt{blacken} does not change the active partition of $s$, according to assumption 1 we have

\[
\text{select} (\text{next} (s), a) = \text{select} (\text{next} (\text{blacken} (s)), a).
\]

\textbf{hence select} (\text{next} (s), a) = \text{select} (\text{next} (\text{blacken} (s)), a) \\
\text{by (metis 1 B2 B3 equals_def)}
\]

But of course \texttt{blacken}$(s)$ is black for all segments; by B5, so is \texttt{next} (\texttt{blacken}$(s)$). Because by B4 blackness is a function of the contents of a memory segment, we get

\[
\text{thus black} (\text{next} (s), a)
\]

\textbf{by (metis B1 B4 B5)}

which completes the proof.

Unfortunately, the \textbf{Black} axiom is not sufficient to show the FW\_Correct security requirement. This problem is the topic of the next section.

6. PRESERVATION OF BLACKNESS

In the paper introducing the \textbf{Black} axiomatisation of the \textbf{black} predicate[2], Van der Meyden appears to prove that together with the FW\_Pol, FW\_Blackens, and \textbf{Separation} axioms, the \textbf{Black} axiom is sufficient to prove the FW\_Correct theorem specifying the secure operation of the Firewall. Unfortunately, this proof is incorrect, and the \textbf{Black} axiom is in fact not strong enough to ensure that the \textbf{black} predicate is sufficiently well-behaved.

To prove FW\_Correct, Van der Meyden correctly shows that partitions other than $B$, including the Firewall partition $F$, can never make any segments of $B$ nonblack. He also shows that $B$ can never make any segments other than \texttt{outbox} nonblack without some other segment already being nonblack. A problem occurs, however, in proving that $B$ can never make \texttt{outbox} nonblack. For this to be the case due to the \textbf{Black} axiom, there needs to be a set of segments $X$ such that the next contents of \texttt{outbox} is a function of the active partition and the contents of the segments in $X$.

Van der Meyden shows that a set of segments $X$ exists such that among all states $s$ for which \texttt{current}$(s) = B$, the contents of \texttt{outbox} in \texttt{next}$(s)$ is a function of the contents of the segments in $X$. That is, he constructs a set $X$ such that for all segments $r$ and $t$ for which \texttt{current}$(r) = \texttt{current} (t) = B$, if \texttt{equals}$(X, r, t)$ holds, then we can conclude \texttt{select} (\texttt{next}$(r)$, \texttt{outbox}) = \texttt{select} (\texttt{next}$(t)$, \texttt{outbox}). Based on the “no spontaneous generation of nonblack data” intuition, we would expect this to be sufficient to show that \texttt{black} (\texttt{next}$(s)$, \texttt{outbox}) holds under the assumption that \texttt{current}$(s) = B$ and all segments in $X$ are black in $s$, and Van der Meyden argues exactly this in his proof of FW\_Correct. This does not follow from the \textbf{Black} axiom, however.

Indeed, there is nothing in the \textbf{Black} axiom that requires a computation step of $B$ to maintain the blackness of \texttt{outbox} when all segments accessible to $B$ are black. This would require the next contents of \texttt{outbox} to be a function of the active partition and \texttt{segs}(B). But in general, this is not the case; the Firewall partition generally writes to \texttt{outbox} based on segments not accessible to $B$, which means that the next content of \texttt{outbox} is not independent of those segments. Appendix A describes a specific counterexample in which this is the case, showing that the \textbf{Separation}, FW\_Pol, FW\_Blackens, and \textbf{Black} axioms can all be true while FW\_Correct is false.

To remedy this, we propose a stronger version of the \textbf{Black} axiom that does not suffer from this problem, which we feel better formalises the intuition behind the \textbf{Black} axiom. In his flawed proof, Van der Meyden inadvertently argues that the next contents of \texttt{outbox} are a function of the active partition and the contents of a set $X$ of segments \textit{among those states $s$ for which \texttt{current}$(s) = B$}; because this predicate \texttt{current}$(s) = B$ is true for the specific state he is considering, he concludes that blackness follows for \texttt{outbox} in the next state of this specific state. We feel this line of reasoning should hold for \texttt{black} for arbitrary predicates of $s$. That is, if the functionality of segment $a$ on segments $X$ property holds for all states matching some predicate $P$, and all segments of $X$ are black in a state $s$ also matching this predicate $P$, then $a$ should be black in the next state of $s$.

Formally:

\[
\begin{aligned}
\text{assumes StrongBlack:} \forall P \in \mathbb{P} (\text{State}), \\
X \in \mathbb{P} (\text{Segment}), s \in \text{State}, a \in \text{Segment}. \\
(\forall r, t \in \text{Segment} \cdot P (r) \land P (t) \land \\
\text{equals} (X, r, t) \land \\
\text{current} (r) = \text{current} (t) \rightarrow \\
\text{select} (\text{next} (r), a) = \text{select} (\text{next} (t), a)) \\
(\forall b \in X. \text{black} (s, b)) \\
P (a) \\
\text{black} (\text{next} (s), a)
\end{aligned}
\]

This definition is a generalisation of \textbf{Black}; using $P (s) = \text{true}$ yields \textbf{Black} as a special case. Using this stronger ax-
The WeakBlack axiom is a special case of the StrongBlack axiom produced by substituting the predicate $P(t) \equiv current(s) = current(t)$ for the variable $P$; thus, it trivially follows from StrongBlack. More interestingly, it also follows from Rushby’s axioms, using almost exactly the same proof as the one in Section 5.3:

```isar
fix X s a
assume 1: 
(∀r,t ∈ Segment.
current(s) = current(t) ∧ current(r) = current(t) ∧
equals(X, r, t) →
select(next(r), a) = select(next(t), a))
assume (∀b ∈ X.black(s, b))

by (metis 1 B2 B3 equals_def)

thus black(next(s), a)
by (metis B1 B4 B5)
```

When using the WeakBlack axiom instead of Black, Van der Meyden’s proof is correct. We prove this below by presenting Van der Meyden’s proof in fully formalised form in the Isabelle proof system.

The theorem we want to prove is that FW_Correct holds under the assumption of the Separation, FW_Blacks, FW_Pol, and WeakBlack axioms:

```
theorem
assumes Separation: ∀s, t ∈ State, a ∈ Segment.
equals(dia(a) ∩ segs(current(s)), s, t) ∧
current(s) = current(t) ∧
select(s, a) = select(t, a) →
select(next(s), a) = select(next(t), a)

assumes FW_Pol: ∀a, b ∈ Segment, P ∈ Partition.
a ∈ segs(B) ∧
b ∈ dia(a) ∧
b ∈ segs(P) ∧
P ≠ B →
(P = F ∧ a = outbox)

assumes FW_Blacks: ∀s ∈ State.
current(s) = F ∧ black(s, outbox) →
black(next(s), outbox)

assumes WeakBlack: ∀X ∈ P(Segment), s ∈ State, a ∈ Segment.
(∀r,t ∈ Segment.current(s) = current(r) ∧
equals(X, r, t) ∧
current(r) = current(t) →
select(next(r), a) = select(next(t), a)) ∧
(∀b ∈ X.black(s, b)) →
black(next(s), a)
```

Proof of this property is ultimately by induction on $n$. To simplify things, we first prove the correctness of the induction step as a lemma.

```
proof -
```
have 0: \( \forall s \in \text{State}, a \in \text{Segment}. \) 
\( (\forall b \in \text{segs}(B). \blacken{(s, b)}) \rightarrow \) 
\( a \in \text{segs}(B) \rightarrow \blacken{(\text{next}(s), a)} \)

proof -
fix \( s, a \)
assume 1: \( \forall b \in \text{segs}(B). \blacken{(s, b)} \)
assume 2: \( a \in \text{segs}(B) \)

We now need to prove that \( \blacken{(\text{next}(s), a)} \). This proof will proceed by cases, but first we prove the simple lemma that \( \text{FW}_{-}\text{Pol} \) applies to \( a \): in other words, that if something can influence \( a \), then \( a \) must be \( \text{outbox} \) and that something must be accessible only to \( B \) and \( F \). This is a triviality, but proving it here will spare us the effort of having to duplicate the proof in all the cases that make use of it.

with \( \text{FW}_{-}\text{Pol} \) have 3:
\( \forall b \in \text{Segment}, P \in \text{Partition}. \)
\( b \in \text{segs}(P) \rightarrow P \neq B \rightarrow b \in \text{dia}(a) \rightarrow (a = \text{outbox} \land P = F) \)
by simp

The proof of \( \blacken{(\text{next}(s), a)} \) will proceed by cases. We first consider the case where \( a \neq \text{outbox} \).

show \( \blacken{(s, a)} \)
proof cases
assume 4: \( a \neq \text{outbox} \)

Because \( a \neq \text{outbox} \), by \( \text{FW}_{-}\text{Pol} \) and \( \text{Separation} \) it follows that the next contents of \( a \) are a function of the contents of \( \text{segs}(B) \) and the active partition. The \( \text{WeakBlack} \) axiom then requires that the blackness of the segments in \( \text{segs}(B) \) which we assumed in assumption 1. To show this, we first need to establish the functional dependence of \( a \) on \( \text{segs}(B) \) and the active partition as a lemma to later feed to \( \text{WeakBlack} \).

have \( \forall r, t \in \text{State}. \)
\( \text{current}(s) = \text{current}(r) \rightarrow \) 
\( \text{current}(r) = \text{current}(t) \rightarrow \) 
\( \text{equals}(\text{segs}(B), r, t) \rightarrow \) 
\( \text{select}(\text{next}(r), a) = \text{select}(\text{next}(t), a) \)
proof auto
fix \( r, t \)
assume 5: \( \text{current}(s) = \text{current}(t) \)
assume 6: \( \text{current}(r) = \text{current}(t) \)
assume 7: \( \text{equals}(\text{segs}(B), r, t) \)

Because \( a \in \text{segs}(B) \) and \( \text{equals}(\text{segs}(B), r, t) \), we have \( \text{select}(r, a) = \text{select}(t, a) \).
with 2 have 8: \( \text{select}(r, a) = \text{select}(t, a) \)
unfolding \text{equals}_{-}\text{def} \) by simp

Because of \( \text{FW}_{-}\text{Pol} \) and the fact that \( a \neq \text{outbox} \), we must have \( \text{dia}(a) \subseteq \text{segs}(B) \). Because \( \text{equals}(\text{segs}(B), r, t) \) and \( (X \cap Y) \subseteq X \), we certainly have the following:

from 3 4 7 have \( \text{equals}(\text{dia}(a) \cap \text{segs}(\text{current}(r)), r, t) \)
unfolding \text{equals}_{-}\text{def} \) by auto

But then \( \text{Separation} \) gives us the desired result:

with 6 8 \( \text{Separation} \) show \( \text{select}(\text{next}(r), a) = \text{select}(\text{next}(t), a) \)
by simp
qed

This finishes the lemma stating that the next contents of \( a \) are a function of the contents of \( \text{segs}(B) \) and the active partition. The \( \text{WeakBlack} \) axiom will now prove the blackness of \( a \) in \( \text{next}(s) \).

with 1 \( \text{WeakBlack} \) show \( \blacken{(\text{next}(s), a)} \) by auto

This finishes the case where \( a \neq \text{outbox} \).

Next, we make a further case distinction on the value of the active partition in \( s \). We consider three cases: \( \text{current}(s) = B, \text{current}(s) = F, \) and \( \text{current}(s) \neq B \land \text{current}(s) \neq F. \)

next assume 4: \( a \neq \text{outbox} \)
show \( ?\text{thesis} \)
proof cases
assume 5: \( \text{current}(s) = B \)
The case where \( \text{current}(s) = B \) uses the same structure as the \( a \neq \text{outbox} \) case. It first proves the lemma that the next contents of \( a \) are a function of the contents of \( \text{segs}(B) \) and the active partition.

have \( \forall r, t \in \text{State}. \)
\( \text{current}(s) = \text{current}(r) \rightarrow \) 
\( \text{current}(r) = \text{current}(t) \rightarrow \) 
\( \text{equals}(\text{segs}(B), r, t) \rightarrow \) 
\( \text{select}(\text{next}(r), a) = \text{select}(\text{next}(t), a) \)
proof auto
fix \( r, t \)
assume 6: \( \text{current}(s) = \text{current}(t) \)
assume 7: \( \text{current}(r) = \text{current}(t) \)
assume 8: \( \text{equals}(\text{segs}(B), r, t) \)
with 2 have 9: \( \text{select}(r, a) = \text{select}(t, a) \)
unfolding \text{equals}_{-}\text{def} \) by simp

The details of this step are different from the \( a \neq \text{outbox} \) case, however. Here, the next contents of \( a \) are only a function of the contents of \( \text{segs}(B) \) and the active partition for those states \( s \) with \( \text{current}(s) = B \); in other words, this is the point in the proof in which the difference between \( \text{WeakBlack} \) and \( \text{Black} \) is crucial, and it’s the point where Van der Meyden’s original proof is incorrect. Because \( \text{current}(r) = \text{current}(t) \), we have \( \text{equals}(\text{dia}(a) \cap \text{segs}(\text{current}(r)), r, t) \):

from 5 6 7 8 have
\( \text{equals}(\text{dia}(a) \cap \text{segs}(\text{current}(r)), r, t) \)
unfolding \text{equals}_{-}\text{def} \) by simp

The remainder of this case proceeds in the same way as the case where \( a \neq \text{outbox} \).

with 7 9 \( \text{Separation} \) show \( \text{select}(\text{next}(r), a) = \text{select}(\text{next}(t), a) \)
by simp
qed
with 1 \( \text{WeakBlack} \) show \( \blacken{(\text{next}(s), a)} \) by auto

This concludes the case where \( \text{current}(s) = B \).

The case where \( \text{current}(s) = F \) is almost trivial; blackness of \( a \) in \( \text{next}(s) \) follows immediately from its blackness in \( s \) and \( \text{FW}_{-}\text{Blackens} \):

next assume 5: \( \text{current}(s) \neq B \)
show \( ?\text{thesis} \)
proof cases
assume 6: \( \text{current}(s) = F \)
from 1 2 4 have \( \blacken{(s, \text{outbox})} \) by simp
with 4 6 \( \text{FW}_{-}\text{Blackens} \) show \( ?\text{thesis} \) by simp
All that remains is the case where both $\text{current}(s) \neq B \land \text{current}(s) \neq F$. This, too, is a simple case: the security policy forbids the active partition from modifying any of the segments of $B$, which means the same proof used for the $a \neq \text{outbox}$ case applies.

\text{next assume 6: } \text{current}(s) \neq F \\
\text{have } \forall r, t \in \text{State}. \\
\text{current}(s) = \text{current}(r) \rightarrow \\
\text{current}(r) = \text{current}(t) \rightarrow \\
\text{equals}(\text{segs}(B), r, t) \rightarrow \\
\text{select}(\text{next}(r), a) = \text{select}(\text{next}(t), a) \\
\text{proof auto} \\
\text{fix } r, t \\
\text{assume 8: } \text{current}(s) = \text{current}(t) \\
\text{assume 9: } \text{current}(r) = \text{current}(t) \\
\text{assume equals}(\text{segs}(B), r, t) \\
\text{with 2 have 10: } \text{select}(r, a) = \text{select}(t, a) \\
\text{unfolding equals_def by simp} \\
\text{with 3 5 6 8 9 have} \\
\text{equals}(\text{din}(a) \cap \text{segs}(\text{current}(r)), r, t) \\
\text{unfolding equals_def by auto} \\
\text{with 9 10 Separation show} \\
\text{select}(\text{next}(r), a) = \text{select}(\text{next}(t), a) \\
\text{by simp} \\
\text{qed} \\
\text{with 1 WeakBlack show black(\text{next}(s), a) by auto} \\
\text{qed qed qed qed}

Because this is the last case, this finishes the proof of the lemma which states the correctness of the induction step of the main theorem. That is, we just proved $\forall s, a, (\forall b \in \text{segs}(B). \text{black}(s, b)) \rightarrow a \in \text{segs}(B) \rightarrow \text{black}(\text{next}(s), a)$.

With this lemma in hand, we can now easily prove the main theorem, with an appeal to the lemma 0 in the induction step:

$$
\forall s \in \text{State}, n \in \mathbb{N}.
(\forall a \in \text{segs}(B). \text{black}(s, a)) \rightarrow \\
(\forall a \in \text{segs}(B). \text{black}(\text{run}(n, s), a))
$$

\text{proof -} \\
\text{fix } s, n \\
\text{assume } \forall a \in \text{segs}(B). \text{black}(s, a) \\
\text{then show } \forall a \in \text{segs}(B). \text{black}(\text{run}(n, s), a) \\
\text{proof (induction } n, \text{ auto)} \\
\text{fix } n, x \\
\text{assume 2: } \forall x \in \text{segs}(B). \text{black}(\text{run}(n, s), x) \\
\text{assume 3: } x \in \text{segs}(B) \\
\text{with 0 2 show black(\text{next}(\text{run}(n, s)), x) by simp} \\
\text{qed qed qed qed}$$

This completes the proof.

8. CONCLUSION

In the previous section, we have proven the formal property $\text{Separation} \land \text{FW_Pol} \land \text{FW_Blackens} \land \text{WeakBlack} \rightarrow \text{FW_Correct}$. That is, we have shown that the desired security property that the untrusted application does not gain access to unprivileged information holds, under assumptions that the separation kernel and Firewall application behave in a certain way.

The point of this exercise is to study techniques for the formal verification of system properties in a compositional way. In this paper we did not prove any properties about the behaviour of system components such as the separation kernel or the Firewall; instead, these properties were taken for granted. The problem studied in this paper is how to make use of independently proven properties describing individual system components to prove properties of the system as a whole in which these components play a role; that is, to compose verified behaviour of components into verified behaviour of the complete system.

When verifying practical systems, one would presumably independently prove behavioural properties regarding individual system components, and then later attempt the composition in a way similar to the methods used in this paper. Based on our experience formalising the notions presented in this paper, we feel confident that compositional formal validation of system properties is a practical technique for certifying desired system properties in applications such as security certification.

Acknowledgments

We acknowledge funding from the European Union’s Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 318353 (EURO-MILS project: http://www.euromils.eu).

9. REFERENCES

APPENDIX

A. COUNTEREXAMPLE TO VAN DER MEYDEN’S AXIOM

In Section 6, we described how a system can be constructed that satisfies the Separation, FW_Pol, FW_Blackens, and Black axioms, while still not satisfying FW_Correct. For the sake of completion, we provide here a specific minimal system for which this is the case.

Consider a GWV system as described in Section 3 consisting of two segments outbox and inbox, a partition B with \( \text{segs}(B) = \{\text{outbox}\} \), and a second partition F with \( \text{segs}(F) = \{\text{outbox, inbox}\} \). The dia function does not constrain any types of influence: \( \text{dia}(a) = \{\text{outbox, inbox}\} \) for both values of a. The system has three possible states, named \( S_1 \), \( S_2 \), and \( S_3 \). The three states succeed each other in a cycle: \( \text{next}(S_1) = S_2 \), \( \text{next}(S_2) = S_3 \), and \( \text{next}(S_3) = S_1 \).

The contents of the memory, its sensitivity, and the active partition are summarized in Table 1. The \( F \) partition is active in the states \( S_1 \) and \( S_2 \), and \( B \) is active in \( S_3 \). The contents of outbox are equal for states \( S_1 \) and \( S_2 \), and different for \( S_3 \); the contents of inbox are equal for \( S_1 \) and \( S_3 \) and different for \( S_2 \). Of course, the exact values are irrelevant.

The outbox segment is black in states \( S_2 \) and \( S_3 \); the inbox segment is never black. This has the curious property that the states \( S_1 \) and \( S_2 \) have the same contents for the outbox segment, but differing blackness for that segment: the GWV and Rushby axiomatisations of the black predicate would not allow this, but it is possible under the Black axiom.

The system described here satisfies the Separation axiom. Because the dia function describes the complete relation and thus does not forbid anything, and because \( B \) does not write to inbox in the one state in which it is active, this is trivial.

Because \( B \) and \( F \) are the only partitions in the system and outbox is the only segment in segs(B), FW_Pol is trivially satisfied. The FW_Blackens axiom is easily checked by verifying that black(next(s),outbox) is true for every state s with current(s) = \( F \).

Showing that this system satisfies Black is less obvious. The value of outbox in the next state of s is not a function of the current value of outbox and the active partition; indeed, the states \( S_1 \) and \( S_2 \) have the same active partition and the same values for outbox, yet select(next(S_1),outbox) = 1 \( \neq \) 2 = select(next(S_2),outbox). The value of outbox in the next stage of s is a function of the current value of inbox and the active partition, and therefore also of the superset {outbox, inbox}. However, all we can conclude from this using the Black axiom is that if all segments in {inbox} are black, then outbox must be black in the next state. Because inbox is never black, this is vacuously true. Thus, the system satisfies Black in a vacuous way.

The WeakBlack axiom, and therefore also the StrongBlack axiom, would note that among all states for which current(s) = \( B \), the contents of outbox in the next state is a function of the active partition and the contents of outbox; indeed, it is even a function of the active partition and the contents of \( \emptyset \). Thus, they would require that whenever outbox is black in a state for which \( B \) is the active partition, then outbox must still be black in the next state. This system, then, does not satisfy the WeakBlack axiom, and is not a counterexample against it.

For the Black axiom, however, it is indeed a counterexample. For this system does not satisfy FW_Correct; in state \( S_1 \) all segments of \( B \) are black, yet in the next state \( S_1 \), this is no longer the case. That makes this a system for the Separation, FW_Pol, FW_Blackens, and Black axioms are satisfied, yet FW_Correct does not hold, and a proof that Van der Meyden’s proof is flawed.

<table>
<thead>
<tr>
<th>s</th>
<th>next(s)</th>
<th>current(s)</th>
<th>select(s,outbox)</th>
<th>select(s, inbox)</th>
<th>black(s,outbox)</th>
<th>black(s,inbox)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S_1</td>
<td>S_2</td>
<td>( F )</td>
<td>1</td>
<td>3</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>S_2</td>
<td>S_3</td>
<td>( F )</td>
<td>1</td>
<td>4</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>S_3</td>
<td>S_1</td>
<td>( B )</td>
<td>2</td>
<td>3</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

Table 1: The Firewall MILS Example.