This paper was rejected by ACSAC 2010, but we think it worth making available.

An Application of the MILS Approach To Secure Information Sharing

Rance DeLong
LynuxWorks
San Jose CA

David Hanz* and John Rushby
SRI International
Menlo Park CA 94025 USA

November 2009

Abstract

MILS is an approach to the design of secure systems that supports modularity. MILS protection profiles encourage development of a COTS marketplace for trusted components, and the MILS principles for compositional assurance then allow assurance for the full system largely to be derived from that of its components. We illustrate the MILS principles for secure systems design and assurance through an example in which they are applied to a real system being developed to support military training in coalition operations.

1 The MILS Approach

MILS is an architectural approach to the design and assurance of secure systems. MILS-like systems support the DDG-1000 Zumwalt-class destroyer and the next-generation display system for all ships in the US Navy, and they provide the avionics for the F-35 and the second-generation avionics for the F-22 fighter planes.

MILS was originally an abbreviation for "Multiple Independent Levels of Security" but it is now best considered simply a name. Earlier interpretations of MILS focus on an approach to secure systems design composed of three layers: at the bottom, physical distribution and separation kernels [4] are used to partition resources; next, a layer of trusted "middleware" manages these to provide secure services to applications (usually untrusted, and possibly running on legacy operating systems), which comprise the top layer [1]. More recent interpretations of MILS focus on a two-level approach that supports compositional (i.e., modular) security assurance [2]. The two interpretations are perfectly compatible: the earlier one concentrates on the structural aspects of design, the later one takes a more abstract,

*Sadly, Dave Hanz died in April 2020. We dedicate this paper to his memory.

logical view. In this paper, we illustrate the latter interpretation of MILS by showing how its principles can be used to guide the development of an architecture to support military training in coalition operations.

By the MILS principles, we mean the precepts for secure systems architectures presented in [2]. These may be summarized as follows.

- Security is often an important attribute of a system, but seldom its primary functional purpose. Enforcing security should not significantly increase the cost of the system nor degrade its ability to achieve its purpose. Hence, the goal of a MILS architecture is to facilitate achieving the functional *purpose* of the system while simultaneously enforcing its *security policy*.

- A MILS architecture is composed of *trusted* and *untrusted* components. The functional purpose of the system is generally achieved by untrusted components, while trusted components enforce security by protecting and enforcing the interfaces to untrusted components, and by mediating information flows among them. Untrusted components may be COTS or may be developed specifically for the system concerned.

- Trusted MILS components may be divided into those that are foundational and those that are operational. *Foundational* components work "behind the scenes" to protect and enforce the interfaces to untrusted components; they may also provide the capability to multiplex several trusted or untrusted components onto a single physical resource, in which case they are called *resource-sharing* foundational components. Examples of the latter include separation kernels and partitioning file systems and networks.

- Separation kernels allow untrusted components to be replicated at essentially zero cost. It is often possible to construct a secure system by replicating the untrusted applications that accomplish the purpose of the system so that each replica operates at a single security level; trusted *operational* components then mediate the "cross domain" flows of information between the levels.

- Unlike foundational components, trusted operational components perform some visible security-relevant function: they may mediate the flow of information between separate security domains, or they may perform some *multilevel* function. The latter kind of component is often internally structured as a MILS system in its own right (e.g., a multilevel file system may be built as a mediating front end to a partitioning file system) and as an element in a larger MILS system it is then referred to as *compound* operational component.

- MILS supports a COTS-like business model through sponsorship of Common Criteria *Protection Profiles* for standard kinds of MILS trusted components,

such as separation kernels, partitioning file systems and networks, and guards. Many suppliers can then compete to deliver products conformant with these Protection Profiles.

- The untrusted components and the operational trusted components constituting a system design, their interfaces, and the (unidirectional) paths for information flow among them can be represented in a "boxes and arrows" diagram called a *policy architecture*. The policy architecture is the interface between the two levels of design and assurance in MILS: the foundational trusted components *implement* the policy architecture, and the operational trusted components *rely* on it.

- The responsibility of the trusted foundational components is to provide the resources (boxes) and information flows (arrows) defined in the policy architecture and to *guarantee* the absence of any other channels for information flow; this guarantee should make no assumptions about the behavior of untrusted components or the environment.

- The security properties required for the system constitute its *system security policy*; operational trusted components each have a *local security policy*. The security properties of the system can be *calculated* from the policy architecture and the local security policies of the operational trusted components. These calculated properties must imply the required system security policy.

- Assurance in MILS is *compositional*: that is, it is calculated from the separate assurances of its trusted components. Foundational trusted components are assured to provide a fragment of a policy architecture; assurance for the overall architecture is composed from these. Operational trusted components are assured to enforce a local security policy; assurance for the overall system is composed from these and the policy architecture.

The rest of this paper illustrates these principles for MILS secure systems design through their application to a real example.

2 Introduction to a Military Training Example

Our example is based on a system that is currently in development to support military training in the context of coalition battlefield operations. Military battlefield training involves friendly and surrogate enemy forces (known as OPFOR, for OPposing FORces) and their weapons platforms. Some of the forces and weapons platforms are real, and some are simulated. The real weapons platforms do not fire live weapons; instead, their trajectories and effects are simulated and reported to the participants, some of whom will be deemed to be damaged or destroyed. To enable

this combination of reality and simulation, real weapons platforms are augmented with additional hardware and software that interacts with the platform's sensors, weapons, and countermeasures, and also engages with similar hardware and software on other platforms (both friendly and those playing the part of the enemy) to distribute real and simulated information so that the operators of each platform see a realistic engagement on a common battlefield.

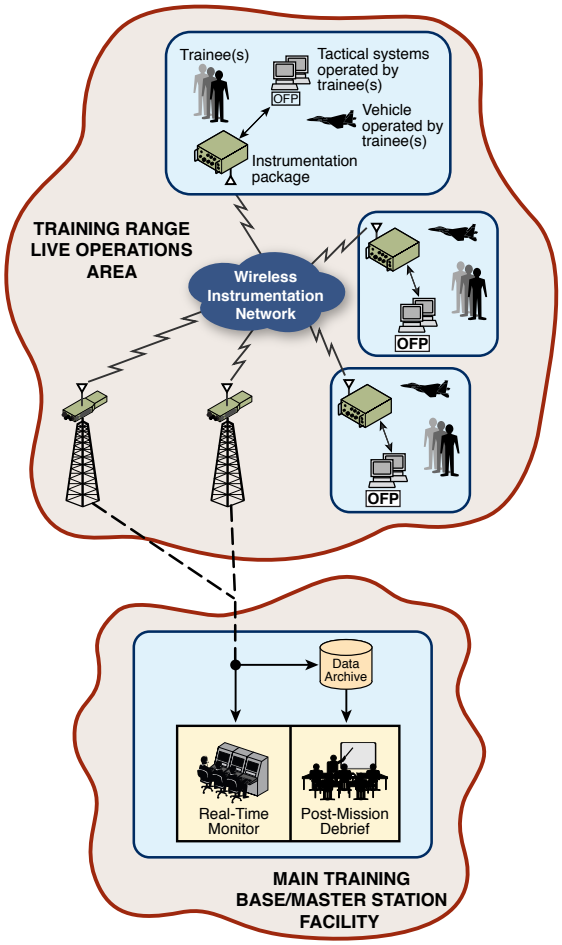

Figure 1: Basic Training Architecture

A picture of the system architecture for live training is shown in Figure 1. In the specific example considered here, the weapons platform is an aircraft and the hardware and software added for training purposes is called the "Training Instrumentation Package," colloquially referred to as the "pod." Within the pod is an "Advanced Data Interface Unit" (ADIU) that runs the training appplication soft-

ware and interacts with the aircraft's sensors and weapons by exchanging messages with the aircraft's "Operational Flight Program" (OFP). The pod also contains a digital radio that provides an IP-based wireless network linking all the pods and their control stations located on the ground. Training application software in the pods installed on all aircraft participating in the training exercise use this network to coordinate their activity.

During an air-to-air engagement, the training application in each aircraft receives messages from the OFP that identifies the targets being tracked by the aircraft's sensors, and any (simulated) weapons being fired; over its pod-to-pod network, it receives the coordinates of other aircraft in the engagement and the weapons they have fired; by correlating sensor tracks with weapons simulations and known aircraft locations, the software can determine a casualty assessment and can supply information to the pilot's displays so that these provide a realistic representation of the engagement.

This basic model of system operation is becoming more complicated to support training for coalitions of forces. Some information about the engagements may be made available to certain members of the coalition but not to others and some information may be made available in full precision to US and certain other forces, but only in degraded form to other members of the coalition. The reasons for this may be to disguise the full capabilities of certain sensor systems, or the fidelity of some OPFOR weapon simulations (which could reveal sensitive intelligence). The simple architecture shown in Figure 1 is now elaborated into that shown in Figure 2, where cryptography is used to partition the communications network into different enclaves that receive different representations of the underlying information; instead of one view of the battlefield, there are now several, with some information being removed or made less precise in the additional views.

The challenge here is to develop an architecture to support this requirement that can be Certified and Accredited with a high level of assurance. We claim that the principles of the MILS approach to secure systems design provide a disciplined method for satisfying this challenge. Moreover, as our development of this example will show, the MILS principles provide a way to identify a preferred choice when multiple design alternatives are available.

In the following sections we describe two candidate MILS architectures for the training example. Both candidates employ many of the same components, but in different configurations. We believe comparison of the two architectures is instructive.

3 First Approach: Downstream Guards

The purpose of the system under consideration is accomplished by the training application software. This is large and untrusted (with respect to security) and

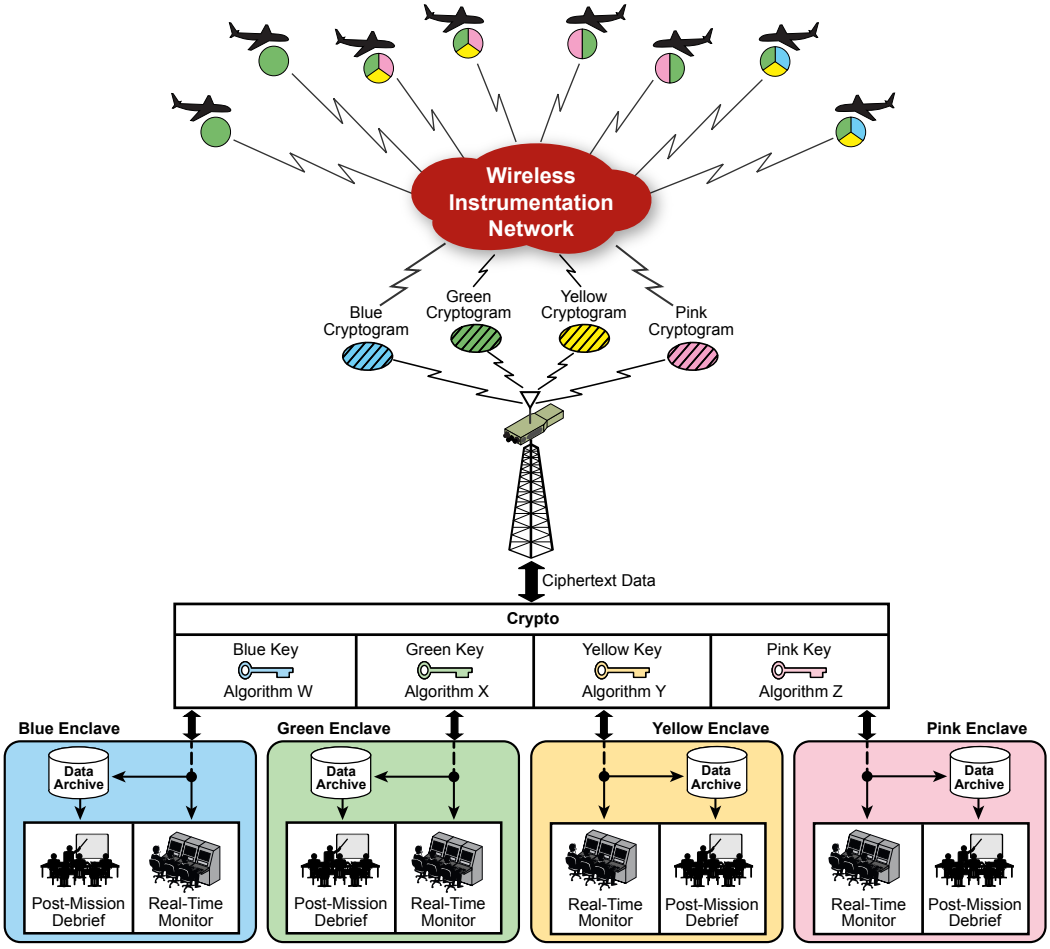

Figure 2: Coalition Training Architecture

our task is to design an architecture that extends the system so that it satisfies the security policy for coalition training exercises. Figure 3 portrays an architecture that has been proposed to achieve this. It is not quite a MILS policy architecture because the information flows are bidirectional, but it is close enough for our purposes.

The right of the Figure shows the Red and Black sides of an SFF-K JTRS radio. JTRS is the *Joint Tactical Radio System* and SFF-K is the "K" variant of the *Small Form Fit* product line. The Red side of the SFF-K contains four physically independent cryptographic units, each of which is connected to a physically independent Ethernet line. The Red side exchanges IP packets (encrypted bodies and plaintext headers) with the Black side that (in the "K" variant) runs the *Range Instrumentation Waveform* (RIW), which provides the functionality of an IP network.

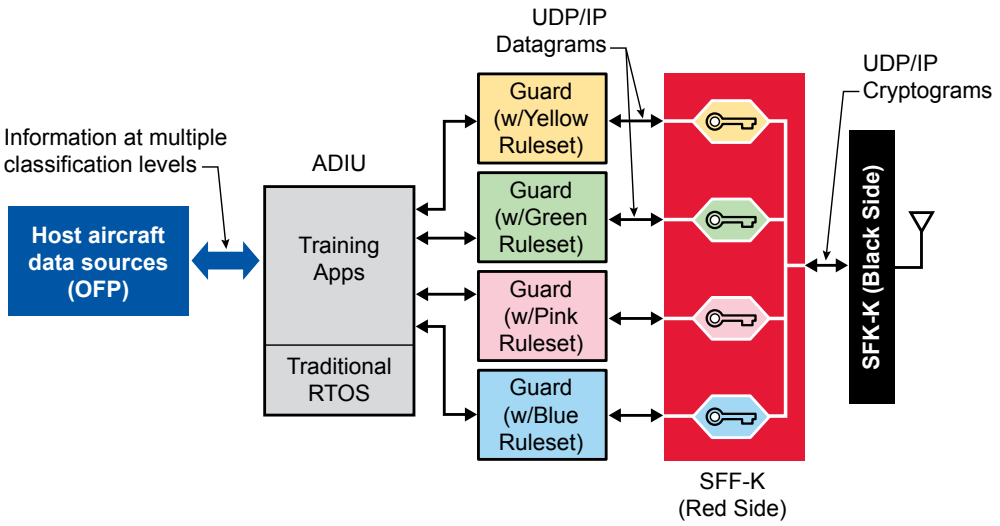

Figure 3: Downstream Guards

The complete SFF-K can be thought of as a MILS system in its own right; here it functions as a MILS *compound* operational component.

The left of the Figure shows the *Operational Flight Program* (OFP), which is a collective term for the software on the aircraft that manages its sensors, weapons, and countermeasures, etc. The *Advanced Data Interface Unit* (ADIU) is the processor in the "pod" that runs the applications software for training exercises. These applications are bespoke software that has been developed over many years; it runs on a conventional real time operating system (RTOS). As described in the previous section, the training applications software receives messages from the OFP providing information from this aircraft's sensors, and any weapons being fired; it also receives messages from the pod-to-pod network providing information about the location of other aircraft and their weapons. The training applications software correlates the information received, simulates weapons that have been fired, determines a casualty assessment, and sends data to the OFP, for presentation on the pilot's displays, and to the radio network, for use by corresponding applications in other aircraft and the control stations.

To support coalition operations, the four channels of the SFF-K are used to provide four cryptographically separated subnetworks; membership of the subnetworks is controlled through cryptographic key distribution. The assignments of specific security enclaves to specific subnetworks can change on a day-to-day basis, but a typical mix might be

- US-only, 5th-Generation Aircraft,

- US-only, all aircraft,
- US and Coalition,
- Coalition-private.

Some of the network data is specific to each subnetwork, but much of it comprises different views of the stream of information exchanged among the training applications; in particular, as the subnetwork membership becomes less selective, some information is omitted and some is reduced in accuracy. This is done to restrict appropriately the distribution of US-classified information.

The training applications cannot be directly connected to the separate channels of the SFF-K because they were developed prior to the coalition requirements and cannot be trusted to segregate different classifications of information. One possible approach would be to re-engineer the training applications so that they are cognizant of this requirement, but this is likely to require substantial effort, and certification at high assurance would be prohibitively difficult and costly. It is fundamental that the cost of assurance increases with the size and complexity of the artifact concerned, and the training applications are large and complex, comprising many hundreds of thousands of lines of code. Instead, the proposal is to interpose a MILS mediating operational component called a *guard* between the training applications and each channel of the SFF-K. The guards could each be implemented as independent appliances or, more likely, they would each run in their own partition of a separation kernel that serves as a MILS *resource-sharing* foundational component.

The guards are transparent to information flowing in the right-to-left (SFF-K to ADIU) direction. Each aircraft has cryptographic keys loaded into its SFF-K that give access only to its allowed subnetworks. The training applications therefore have available only information from their own aircraft and from allowed subnetworks and may safely compute the information to be presented to the aircraft's displays.

In the left-to-right direction, the guards examine information from the training applications and remove or modify (e.g., by reducing precision) selected data fields according to the security policy for the subnetwork concerned.

One argument against this approach is that, in order to identify the fields to be removed or modified, the guards must have knowledge of the format or labeling of messages sent to the network by the training applications, and they must *trust* that knowledge. If the format or labels are different than assumed—either by accident or by malevolent design—the guards will be deceived and may release sensitive information. Hence, this architecture implicitly trusts the training applications to label and format data suitably—but the whole reason for using guards in the first place is to eliminate trust in the training applications.

A counterargument is that it is essential to achieving the purpose of the system that the distributed training applications are able to interpret messages correctly;

hence if message formats or labels were other than specified, the system would not operate correctly. This argument might be plausible if the training applications will remain unchanged from their pre-coalition form. However, it is likely that the training applications will be subject to constant maintenance and improvements and it will be very difficult to guarantee that sensitive information is not encoded into an innocuous data field once the architecture of the coalition training system becomes known.

A more compelling argument against this approach is that unless all the internal information flows of the training applications are known, it is impossible to tell what source information has been used in the construction of any given output, and hence it is difficult to deduce its true sensitivity. For example, an output field might report its own aircraft's estimate of the position of another. This estimate might be fused from several different sensors and simply reducing its precision may not adequately reduce its sensitivity: the fact that the other aircraft is seen at all may reveal important information about one of the sensors. Thus, unless we know what sensors feed into each calculation we cannot adequately sanitize their outputs.

Similarly, it may be insufficient to simply sanitize outputs of the weapons simulations performed by the training applications: different classifications of outputs may need to be generated by fundamentally different simulations, and it will be difficult to segregate the different simulations and their information flows within untrusted applications programs. Even if the different simulations are performed by different subprograms, their strict separation cannot be guaranteed by an untrusted conventional operating system.

The inescapable conclusion is that security of the architecture of Figure 3 cannot be guaranteed in the absence of fairly strong assurance about the internal information flows and other security-relevant properties of the training applications. Such assurance is likely to be impossible or expensive to develop, and will complicate future modifications to these applications. Thus, we deduce that the proposed architecture violates some of the precepts of MILS: namely, that security should not greatly complicate the software that achieves the purpose of the system (e.g., by requiring it to be trusted and therefore assured). Hence, this architecture cannot be recommended.

4 Preferred Approach: Upstream Guards

We now turn to an alternative architecture for coalition training exercises that is portrayed in Figure 4. In this architecture, the "downstream" guards between the training applications and the radio network are moved to an "upstream" location between those applications and the Operational Flight Program, and the training applications are replicated. A separation kernel partitions the replicas from each other; it is likely that the upstream guards could also run as partitions on the same

separation kernel. Colors (Yellow, Green, Pink, and Blue) are used here to denote the different independent security domains.

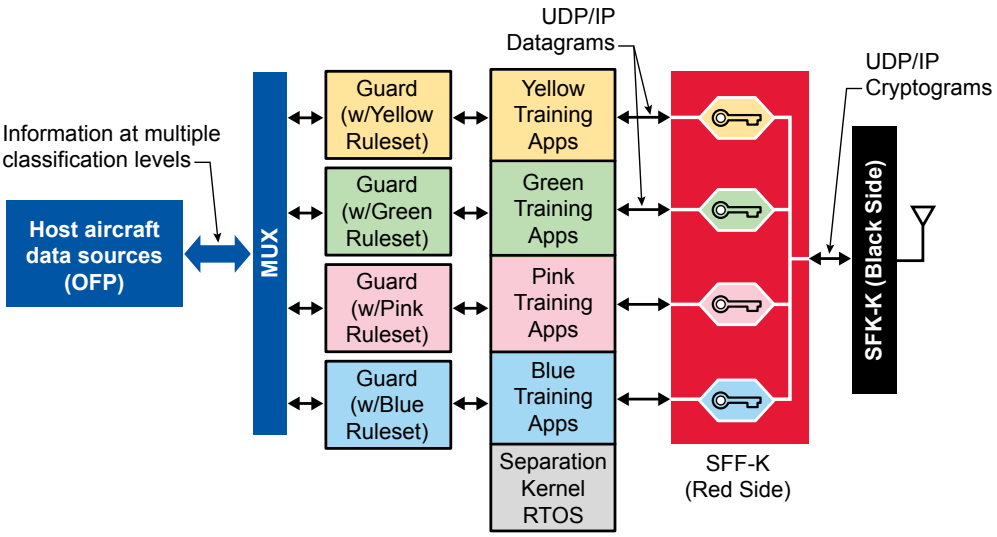

Figure 4: Upstream Guards

As before, each aircraft has cryptographic keys loaded into its SFF-K that give access to only the subnetworks allowed for that coalition member. If we assume that the coalition information classifications are properly nested, then one of the subnetworks to which the aircraft has access provides the maximal network information available to it. For the right-to-left information flow, the guard of the training partition attached to that subnetwork is transparent, and others are closed. In this way, the maximal information is made available to the OFP for presentation on the pilot's displays. If the classifications are not properly nested, then some fusion across multiple partitions may be needed; an augmented architecture to accomplish this is presented later.

In the left-to-right direction, the guards examine information from the OFP and remove or modify (e.g., by reducing precision) selected data fields according to the security policy for the coalition channel concerned. Each guard supplies information to a replica of the training applications suite dedicated to that coalition channel. That replica of the training applications therefore has access only to appropriate information from the OFP and from the network (this is ensured by the cryptographic key loaded into the SFF-K channel to which it is attached), and can therefore be untrusted.

An argument against this architecture is that the upstream guards depend on the formats and labels employed by the OFP in just the same way as the downstream guards of the previous architecture depend on the formats and labels employed by

the training applications. And just as we were forced to conclude that the previous architecture required the training applications to be trusted, so this one requires the OFP to be trusted.

There is some merit to this argument, but differences between the OFP and the training applications weaken it significantly. It is true that the present architecture depends on the format and labels of information output by the OFP in much the same way as the previous architecture depends on the training applications, but whereas the training applications are complex, bespoke programs, subject to modification and enhancement, the OFP is more like part of the aircraft: many other systems depend on it (so it is stable) and its messages are fairly direct presentations of interfaces to the aircraft's subsystems. Furthermore, because it is so vital to the aircraft's operation, it is subjected to rigorous scrutiny and testing. Thus, we argue that although the present architecture depends on the data formats and labels of the OFP, these are so fundamental that it is reasonable to trust them. If this argument is rejected, then the OFP must be replaced by a version for which appropriate assurance can be developed. This would be expensive, but far less expensive than developing comparable assurance for the training applications (which in this case would still depend on an untrusted OFP).

As described for the previous architecture, it is possible that different coalition channels should employ different weapons simulations, rather than the same simulations operating on different data. For this case, the training applications must be under trusted configuration management to ensure that the replica in each partition has only the appropriate simulation code installed. Because the replicated training applications run in partitions of a separation kernel, an attractive option is to move the simulations to their own partitions, where they can be accessed via the inter-process communication mechanisms of the separation kernel. This would move trust from configuration of the (otherwise untrusted) applications programs to the (already trusted) configuration of the partitions and channels of the separation kernel. The single suite of training applications employed in the first architecture cannot exploit this attractive arrangement.

5 MILS Policy Architecture for Preferred Approach

The previous section described an architecture for coalition training based on replicated training applications and upstream guards, and explained why this architecture is preferred to the alternative with downstream guards presented earlier. In this section, we elaborate the preferred approach a little, and present it in the form of a policy architecture with attendant global and local policies and an outline of its assurance argument.

A policy architecture for the preferred approach is presented in Figure 5. This architecture differs from the informal design presented in Figure 4 in that it *is* a

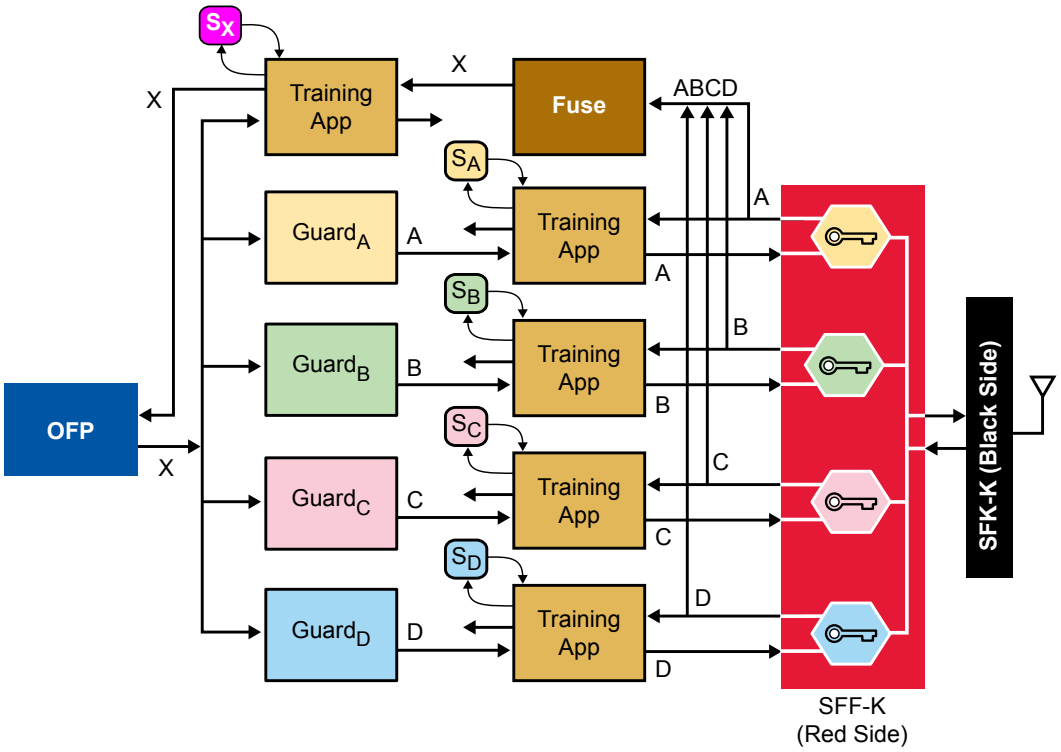

Figure 5: Policy Architecture for Coalition Training

MILS policy architecture (e.g., its information flows are unidirectional), it separates the weapons simulations from the training applications, and it allows coalition classifications that are not properly nested.

By "not properly nested," we mean that there may be some information at coalition classification A that is "better" than the corresponding information at classification B, and there may be other information for which the reverse is true; furthermore, there may be participants X that have access to both A and B who must therefore fuse both sources to obtain the best information they are entitled to access. To accommodate this, we replicate the training applications five times rather than four: four are used in the left-to-right (OFP to network) direction (one for each classification), while the fifth is used in the right-to-left (network to OFP) direction. To keep the fifth replica unmodified, we allocate the fusing of network information to a new, untrusted component, shown as FUSE (it is untrusted because the cryptographic keys loaded into the SFF-K ensure that the only active incoming information flows will be those from allowed coalition channels). This extension of the architecture to coalition classifications that are not properly nested may not be necessary (nor even realistic), but it does illustrate why information flows are

unidirectional in a MILS policy architecture. Because we have separated the right-to-left and the left-to-right information flows, some of the outputs from the training applications are shown as unterminated. In practice, it may be necessary to provide active "sinks" to terminate the protocols on the communications channels concerned. These will be untrusted provided they are encapsulated appropriately (i.e., each a separate "box").

The information flows in Figure 5 are each labeled with a security attribute: these are drawn from A, B, C, and D, representing the four different coalition classifications supported by the SFF-K, and X representing the classification of this aircraft (which may be a union of some of A, B, C, and D).

The local policies of each of the trusted operational components appearing in Figure 5 are the following.

Guard$_A$: input is a sequence of OFP messages of classification X; output is the same sequence sanitized to classification A.

Similarly for Guards$_{B,C,D}$. These are shaded in slightly different colors in Figure 5 to indicate that each guard implements a slightly different policy.

S$_A$ (weapons simulation): input is information of classification A; output is the result of weapons simulation to a fidelity suitable for classification A.

Similarly for S$_{B,C,D}$. These are shaded in slightly different colors in Figure 5 to indicate that each simulation implements a slightly different policy.

SFF-K: this is actually a MILS compound operational component and its internal structure (revealing the red and black sides) would not normally appear in a policy architecture. Its key property is that the four network channels A, B, C, D are securely partitioned. In particular, this means that any information incoming (i.e., in the right to left direction) on channel A *is* of classification A, and similarly for channels B, C, D.

OFP: it is assumed (but not assured) that information sent from the OFP is of classification X.

The Training Applications are untrusted and possibly unmodified from their pre-coalition form. The Fuse function is also untrusted.

The Training Applications, the Simulations, the Guards and the Fuse function could all run on a single computer, each in their own partition provided by a separation kernel serving as a MILS resource-sharing foundational component.

The overall system security policy is

1. Information sent to each subnetwork carries only information appropriate to the classification of that subnetwork

2. Information sent to the OFP (and thus potentially displayed to the pilot and to all others with access to the aircraft communications and storage capabilities) is appropriate for the classification of that aircraft.

The assurance argument for this policy architecture is completely straightforward: untrusted components have access only to information of their own classification (or, in the case of the Fuse function, to information at or below their classification). The trusted guards and Weapons Simulations are assured to deliver information of the appropriate classification. In the case of the guards, part of their assurance will rely on the stability and trustworthiness of the labeling and formatting of data output by the OFP. The trusted SFF-K is assured to properly partition its separate communication channels. The overall assurance argument then follows by simple "geometrical" reasoning based on the topology of the arrows in the policy architecture.

6 Conclusion

We have presented two candidate architectures for a secure training system for coalition operations. The architectures are superficially similar in that both employ concepts from MILS: notably, components that support the *purpose* of the system are left undisturbed and untrusted (but may be replicated), trusted *operational* components act as "Cross Domain" mediators to connect components of different security classification, separation kernels act as trusted *foundational* components to support replication as well as enforce partitioning, and a MILS-based subsystem (the SFF-K radio) is used as a trusted *compound* component.

The architectures differ, however, in the extent to which they truly exploit the architectural opportunities afforded by MILS. The first architecture deploys the existing training applications in what is essentially a multilevel context; trusted guards are then used to separate the thoroughly mixed information flows. This forces the conclusion that the training applications must be trusted and assured, which vitiates the goals of the architecture.

The second architecture applies MILS ideas more vigorously and replicates the training applications for each of the coalition classifications; each replica runs in a dedicated partition provided by a separation kernel. Trusted guards supply each replica with an appropriate view of aircraft information sources provided by the OFP; the SFF-K JTRS radio (itself a MILS system) uses cryptographic separation to connect each replica of the training applications with access to the subnetwork for its coalition classification. Thus each replica of the suite of training applications truly has access only to information of a single classification and may therefore be untrusted.

We concluded that the first architecture does not lend itself to cost-effective development and assurance: its guards are fairly complex and the assurance required of the training applications will be expensive and difficult to develop, and vulnerable to change. In contrast, the second architecture leaves the training applications strictly untrusted, allows trusted configuration for weapons simulations with different fidelity to be managed by the (already trusted) process for configuration of the separation kernel, and requires only simple trusted guards. Furthermore, all the components apart from the OFP and the SFF-K radio can run in partitions provided by a separation kernel running on a single processor board.

Consideration of the functionality and assurance required of the guards in this and similar architectures is a topic worthy of consideration by the MILS community. Almost all MILS applications require guards for "Cross Domain" information sharing in some form or other, and there could be great value in developing a MILS Protection Profile (PP) for these. Developing a generic PP will necessitate creation of a classification scheme for different kinds of guards and for the operations they perform (e.g., removing data fields, reducing accuracy of data in certain fields, and so on). The Unified Cross Domain Management Office (UCDMO) has developed a high-level taxonomy for certain cross domain activities as part of its roadmap [5], but this taxonomy does not provide the level of detail needed to distinguish subtle differences in functionality that are important in applications such as the one described in this paper. Nor do proposals for standardized cross domain protocols (e.g., XMPP) or languages (e.g., Guardol), valuable as those are, address this need—because they focus on syntax, whereas classification must be grounded in semantics. We believe that a classification scheme based on a formal ontology (a semantical notion) for cross domain guard functionality is required for high assurance applications such as these and we urge the MILS community to undertake research on this topic as a necessary precursor to development of a MILS PP for a wide class of guards. An initial version of such a ontology-based classification scheme is described in [3].

References

[1] Jim Alves-Foss, W. Scott Harrison, Paul Oman, and Carol Taylor. The MILS architecture for high-assurance embedded systems. *International Journal of Embedded Systems*, 2(3/4):239–247, 2006.

[2] Carolyn Boettcher, Rance DeLong, John Rushby, and Wilmar Sifre. The MILS component integration approach to secure information sharing. In *27th AIAA/IEEE Digital Avionics Systems Conference*, The Institute of Electrical and Electronics Engineers, St. Paul, MN, October 2008.

[3] Grit Denker, Ashish Gehani, Minyoung Kim, and David Hanz. Policy-based data downgrading: Toward a semantic framework and automated tools to balance need-to-protect and need-to-share policies. In *2010 IEEE International Symposium on Policies for Distributed Systems and Networks*, pages 120–128, 2010.

[4] John Rushby. The design and verification of secure systems. In *Eighth ACM Symposium on Operating System Principles*, pages 12–21, Asilomar, CA, December 1981. (ACM *Operating Systems Review*, Vol. 15, No. 5).

[5] *Cross Domain (CD) Community Roadmap: Building Bridges for Secure Information Sharing*. Unified Cross Domain Management Office (UCDMO), version 1.0 edition, June 2008.