Multiple independent levels of security (MILS) – a high assurance architecture for handling information of different classification levels

Tor Gjertsen
Nils Agne Nordbotten

Forsvarets forskningsinstitutt/Norwegian Defence Research Establishment (FFI)

02.02.2009

FFI-rapport 2008/01999

1086

P: ISBN 978-82-464-1517-8
E: ISBN 978-82-464-1518-5

Keywords

Sikkerhet

Virtualisering

Operativsystem

MILS

MLS

Approved by

Anders Eggen	Project Manager
Vidar S. Andersen	Director

Summary

This report provides an overview of the MILS architecture which is a candidate for high assurance military information systems. Future operational network based concepts set security requirements that are not met by existing solutions. In the search for high assurance platforms, MILS based products were the only found that were designed and planned evaluated for the highest assurance levels. The information in this report is based on open sources found on the Internet.

Sammendrag

Denne rapporten gir en oversikt over MILS arkitekturen som en kandidat til løsning for militære informasjonssystemer med strenge sikkerhets- og tillitskrav. Fremtidige operative nettverksbaserte konsepter stiller sikkerhetskrav som ikke møtes av eksisterende løsninger. I søk på nettet etter sikre plattformer, var MILS-baserte produkter de eneste som kom opp, som var designet for, og hvor det forelå planer om evaluering til de høyeste tillitsnivåene. Informasjonen i denne rapporten er basert på åpne kilder som er tilgjengelig på Internett.

Contents

1 Introduction

Military classified operational systems have had a long standing need for multi-level secure (MLS) operation to increase the flexibility of communications and information systems (CIS). The vision for the NATO Network Enabled Capability (NNEC) as a concept for future military operations has got wide acceptance among NATO nations. As NNEC is a long term vision, not an architecture, the actual NNEC implementations will evolve, and change over time, and so will the security concepts. To be able to make the steps in the right direction, an overall architecture for the NNEC information assurance reflecting the long term vision will be needed.

NNEC is about flexibility and rapid sharing of information in a dynamic environment within a coalition as well as cooperating with civilian governments, contractors and humanitarian organisations. Systems of different classification and belonging to different security domains have to interoperate, and information must be shared securely not to compromise the mission. This is a driving factor for high assurance platforms, and true MLS or MSLS (multi-single level security) capability is becoming more important.

For implementing the ambitions of NNEC high assurance general purpose data platforms are required. That is for different type of equipment, ranging from servers, workstations, clients, guards etc, fixed and mobile and scalable from fixed headquarters down to vehicle and man pack equipment. The introduction of information systems also on the lower operational level will require compact solutions with low power consumption. To reduce space and weight at the lowest levels, one platform should be able to host several systems/applications.

While searching for high assurance platforms suitable for NNEC, the MILS architecture [1;2] (multiple independent levels of security) came up as an interesting alternative for future military operational information systems. This report serves to provide an overview of MILS and its applications. The information in this report primarily originates from open sources found on the Internet.

Past efforts at making computer systems secure usually resulted in higher complexity and cost, as protection mechanisms were added in various layers of the system. The MILS approach is the opposite: to make the protection simpler. Because it is simpler, it can be certified at a higher assurance level.

2 High assurance platforms for NNEC

The implementation of information assurance for NNEC is a huge challenge, and will require new security concepts and solutions. The interconnection of various information domains with different ownership and policies, potentially even civilian, will require high assurance platforms

for critical components. The availability of certified platforms at a sufficient assurance level will be a critical premise for the implementation of NNEC security mechanisms. Some security functions and components will presumably require an assurance level as high as EAL 6-7 according to the Common Criteria [3], depending on the threat scenario and the security span. For others EAL 4-5 may be sufficient.

In fact it is very little “out there” on secure operating systems. On the EAL 5 level there is one general purpose MLS operating system, SXT-400/STOP (LINUX like) from BAE Systems Information Technology. Within this class there are a few more MLS systems at the EAL 4+ level.

However, there is another class of operating systems aimed for high assurance that need attention in the NNEC context; MILS based real time operating systems (RTOS) initially intended for embedded systems. There has been a program in the US initiated by the Air Force Research Lab (AFRL) and the National Security Agency (NSA) to prepare products of the MILS family for use in the Global Information Grid, GIG [4], which is “similar” to NNEC. In addition to AFRL and NSA, the program had participants from research institutions/academia and the industry.

Within this class of operating systems there are several vendors that are expected to have their products evaluated for high assurance. By now, one MILS product has been certified to EAL 6+. This is the INTEGRITY-178B Separation Kernel from Green Hills Software which is listed in the US operated Common Criteria Evaluation and Validation Scheme (CCEVS) [5]. It is not listed in The Common Criteria portal. The reason for this is probably that evaluations at this level are not automatically accepted by other nations, and certificates are therefore only valid nationally.

2.1 What is MILS?

The principles behind the MILS architecture are not new. They were introduced by John Rushby in 1981 in his paper ‘Design and Verification of Secure Systems’ [6]. He proposed that secure systems should be conceived as distributed systems. Security should be achieved partly through physical separation, partly of components and partly through trusted functionality performed within some of these components. The purpose of a security kernel should be to allow such a ‘distributed’ system to run within a single processor. General policy enforcement should not be part of the security kernel. This concept would decouple the verification of the trusted functions in the separated components from the verification of the security kernel. He saw the need for a new verification technique which he called ‘proof of separability’. Application of these techniques should assist the development of systems whose security was based on simple mechanisms and whose verification would be correspondingly simpler than was the case for traditional systems. Rushby did not use the term ‘MILS’. This term is assumed to have been introduced in connection with the US AFRL/NSA initiated MILS program.

The separation kernel concept came first into use in embedded systems that by nature are relatively small, in areas where safety and reliability was vital, like avionics and medical

equipment. It is for safety critical applications in embedded real time systems that these types of products exist today, and they go back to the late 1990's.

Eventually some saw the possibility of using the MILS architecture as a candidate for more general purpose high assurance systems as needed for implementing security critical functions in distributed military information systems for command and control [7], which for this purpose require security evaluation according to Common Criteria (CC).

The basic idea of MILS is to make the security critical part of the system small and with minimum functionality, to make it possible to have it certified for the highest assurance levels (i.e., EAL 6-7). The limited size makes development, certification, and accreditation more practical, achievable, and affordable.

MILS operating systems are not MLS operating systems, but facilitate virtualization with a very strong separation between the partitions (i.e., virtual machines). MILS is different from MLS in the sense that the partitions normally are single level. However the various partitions can be of different levels. A partition can also be MLS, i.e., can contain an MLS application. However, then all the MLS functionality is in the partition itself and not governed by the separation kernel. An MLS partition has to be evaluated as a trusted module, and the supporting middleware/runtime system supporting the application in the MLS partition has to be trusted as well.

2.2 Traditional secure systems

Classified systems in the 70's and 80's were mostly standalone computers with limited physical access. They were typically "system high", and for that mode of operation, access control mechanisms providing need to know separation were sufficient. Increased connectivity and network technology has opened for distributed systems, but they are still operated in the "system high" mode. This often results in several separate networks.

Traditional monolithic operating systems did not incorporate security as a design requirement. Security functions have been added over time, often in the form of "patch and fix" when a security breach has been discovered. This approach is not acceptable for high assurance systems. Furthermore, with the growth of the Internet, the threat picture has changed dramatically and the data systems are far more vulnerable to failures and attacks than at the time the basic design was made.

There have been more serious attempts to bring security into the operating system, like for Multi Level Secure (MLS) versions of UNIX, where critical parts have been redesigned with security in mind. However, they end up being very complex and the security kernel becomes big, which results in a slow and expensive security evaluation process. This has the consequence that they are expensive to maintain and will be lagging behind in the technological evolution. Furthermore, it is not possible to evaluate such monolithic systems to the level required for high assurance systems.

In [8] Randell and Rushby describes the evolution in distributed secure systems, that led to the development of the MILS architecture.

2.3 Security vs. safety

There are well established but different traditions within security and safety even if there are a lot of similarities. Verified security is achieved by Common Criteria (CC) evaluation, with EAL 7 as the highest assurance level. For the higher levels CC certification are based on the use of formal methods, for the highest level this includes the use of mathematical proof.

Safety is governed by RTCA DO-178B requirements with level A as the highest level. Certification is based on the use of traditional development methods and testing. ARINC 653 (Avionics Applications Standards Software Interface) specifies space and time partitioning. An ARINC kernel is similar to a MILS separation Kernel, and each partition has its own memory space. But ARINC is weaker in one aspect; ARINC 653 does not strictly forbid that a partition reads the memory allocated to another partition as is the case for MILS. However, an ARINC based RTOS is well suited as a basis for developing a MILS compliant system.

3 MILS Architecture

The MILS architecture [1;2;4] is layered, and consists of the separation kernel, middleware and the applications. The separation kernel divides the system into separate partitions where the middleware and the applications are located, as shown in Figure 3.1.

In a MILS system traditional OS services like device drivers and file systems runs in middleware. When the MILS architecture is used for virtualization, each virtual machine with its guest operating system runs in a separate partition above the separation kernel. The separation kernel gives a very strong separation between the different partitions/virtual machines.

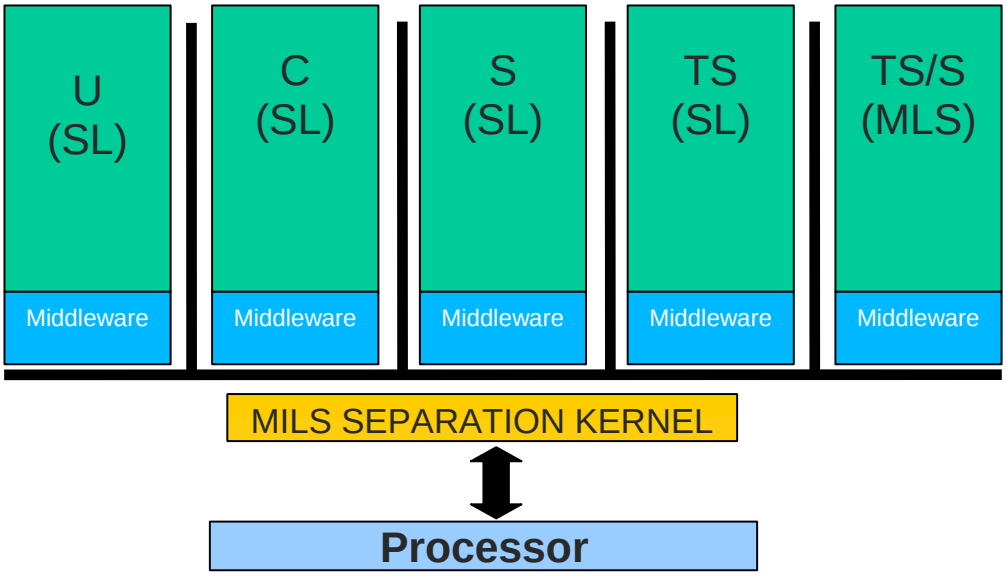


Figure 3.1 The MILS architecture. Notation: Unclassified (U), confidential (C), secret (S), top secret (TS), single level (SL), multi level security (MLS).

3.1 Separation kernel

The separation kernel in a MILS system has a very limited functionality: enforcement of strict separation of partitions (process spaces) and a secure control of information flow between partitions. This makes the separation kernel really small, typically 4K lines of code. That makes the use of formal methods and mathematical verification possible, and enables the exhaustive testing and documentation required for the highest level certification (EAL 6-7).

The MILS separation kernel enforces the following security policy:

- **Data isolation**. Information in a partition is accessible only by that partition, and private data remains private.
- **Control of information flow.** Information flow from one partition to another is from an authenticated source to authenticated recipients, and to nowhere else.
- **Periods processing/sanitization.** The microprocessor and any networking equipment cannot be used as a covert channel between partitions. All shared resources are cleaned before another partition can reuse them.
- **Fault isolation.** Damage is limited by preventing a failure in one partition from cascading to another partition.

It is assumed that middleware and applications have their own policies.

The enforcement of data isolation guarantees that an application in a partition can access only the part of the physical memory that has been allocated to that partition. This counters the threat of compromise, tampering, and virus infection across partitions.

The enforcement of the control of information flow modifies the data isolation policy to allow a limited and well defined provision of information flow between partitions. As an example this flow control can be used to prevent bypass of security functions carried out on the information flow, like encryption.

The enforcement of the periods processing/sanitization policy guarantees that the processor itself is not the means for covert channels that can compromise the system. Before switching to a new partition critical parts are cleared to prevent unintended leak of information between partitions.

The separation kernel is the only part of a MILS system that runs in privileged mode. That means no other code, not even device drivers, has the ability to affect the processor's protection mechanisms. This is a different concept than traditional security kernel based architectures, where a much larger security kernel, with more functionality running in privileged mode, is controlling access to all system resources.

The separation kernel divides the computer into separate address spaces and scheduling intervals, i.e., gives time and space partitioning, and guarantees a strong isolation of the partitions except for the carefully controlled information flows between them. These legal communication paths are configured as part of system set up, i.e., through static configuration.

As an integral part of a MILS product is the trusted initialisation and configuration of the system. This can be implemented in different ways, but the requirements stated in the protection profile for the separation kernel, SKPP [5], require trusted delivery of the configuration data.

3.2 Middleware

The MILS architecture extends the usual concept of middleware. Middleware in the MILS context is reusable software that provides services to the applications.

Most of the traditional OS components like device drivers, file systems, network stacks etc have been moved to middleware, and all run in user mode. It also includes traditional middleware like CORBA and Web services. MILS middleware can be implemented in a number of ways and with varying degree of assurance. A middleware service can be located in a separate partition, or it can reside in the same partition as the application it supports.

3.2.1 Middleware service partitions

Shared resources can be implemented as separate middleware service partitions. Examples of such services are console manager (screen/keyboard/mouse), file system, network stack and so on. Some of these services will have MLS and/or high assurance requirements. MLS will typically provide higher flexibility than single level.

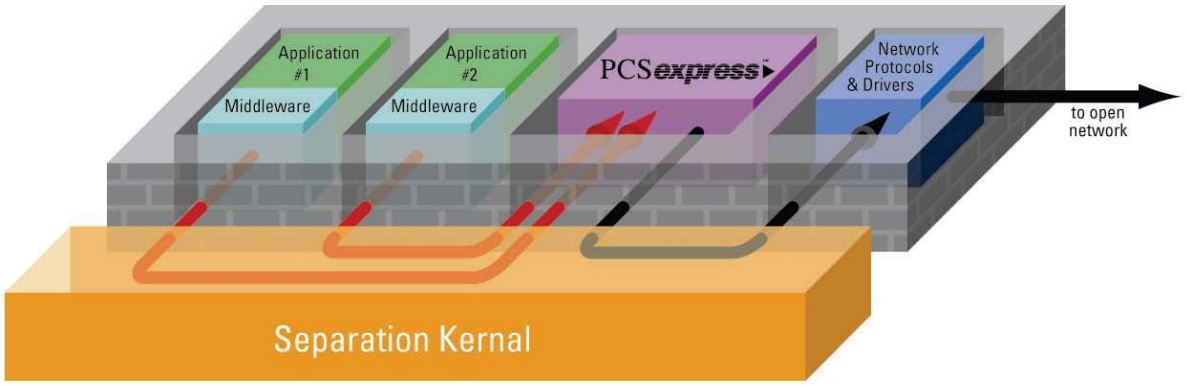
If the system is supported by a security infrastructure, such as a public key infrastructure, the

security service can be implemented in a separate partition. Another example is network based management. In a distributed system with centralized management as part of the system, the partitioning technique can be used to protect the rest of the system from being accessed and manipulated by the management system and vice versa.

3.2.2 PCS

The information flow between partitions can be internally on the same platform or between partitions on different computing platforms. The latter will require end-to-end authentication and integrity, possibly also encryption to maintain data isolation. This functionality is provided by a MILS middleware referred as the Partitioning Communication System (PCS). PCS extends the scope of the separation kernel to inter-system communication. This opens for distributed MILS where one partition can have instances on many physical platforms interconnected via a communication network. PCS is described as part of the MILS architecture [9].

The PCS will be implemented as a separate partition, and the plan is to certify PCS to a high assurance level, ideally the same as the separation kernel. Then the MILS system will maintain its high assurance characteristics also when distributed over a network. A draft PCS protection profile exists and the goal is to have it approved as a standardized protection profile.



The figure is taken from [9] Gordon M. Uchenick, Objective Interface Systems

Figure 3.2 Partitioning Communications System, PCS

3.2.3 Virtualization and guest OS

In order for a standard operating system to be hosted within a partition, virtualization software is implemented as middleware. This way, many guest operating systems can coexist on the same physical platform. Depending on what virtualization software that is available, partitions can contain run time systems for Windows, UNIX, LINUX and others. Because the MILS separation kernel provides a secure separation of the virtual machines, each virtual machine (i.e., guest OS) can run at different security levels [10].

3.2.4 Traditional middleware

MILS specific middleware like PCS will not replace traditional middleware. They can be used in addition, but to be run unmodified, i.e. not MILS adapted, they have to reside in the parturition of the application to be served, and above a virtualized OS corresponding to the middleware implementation. However, a 'traditional' middleware product can be MILS-adapted.

3.3 Applications

An application runs in a MILS partition, above the middleware for that partition. MILS applications manage, control and enforce their own security policies, specified individually for each application, such as firewalls and guards, eventually supported by some middleware services in the same partition or in other partitions. Legacy applications can run unmodified on virtual machines with the actual host operating systems and the proper APIs implemented.

In MILS the different partitions can implement different security policies and reference monitors, and to varying degrees of trust. The separation kernel prevents any side effects from one application to other partitions. A partition can be single level (SL), multiple single level (MSL) or even multi level secure (MLS). In the case of MLS, all the MLS functionality has to be implemented in the partition, and the complete partition has to be evaluated and certified, both the middleware and the application. To implement a specific MLS security function (e.g., a guard as a service in a separate partition) it will typically be implemented on the top of a minimum runtime middleware which, since it is small, is suitable for security evaluation.

A minimum runtime system is typically provided as together with the separation kernel product. The minimal runtime system is typically used to support real-time applications, which were the initial use of the separation kernel architecture in embedded systems.

3.4 MILS workstation

To broaden the use of the MILS architecture to general use, and not limited to embedded systems, the need for a MILS workstation is obvious. Figure 3.3 shows how Ben Calloni of Lockheed Martin sees this, sketched by one of the MILS vendors (Range DeLong, LynuxWorks). It has to include standard middleware components for console, file system, network and security functions in support of a distributed system. Some of these have to operate MLS, while others can operate SL or MSL. How these are chosen will set the properties of the workstation. For instance, the figure shows a multi single level (MSL) file system. In that case the disk will have a static sectioning allocated to the MILS partitions. An MLS file system would be more flexible at the cost of higher complexity.

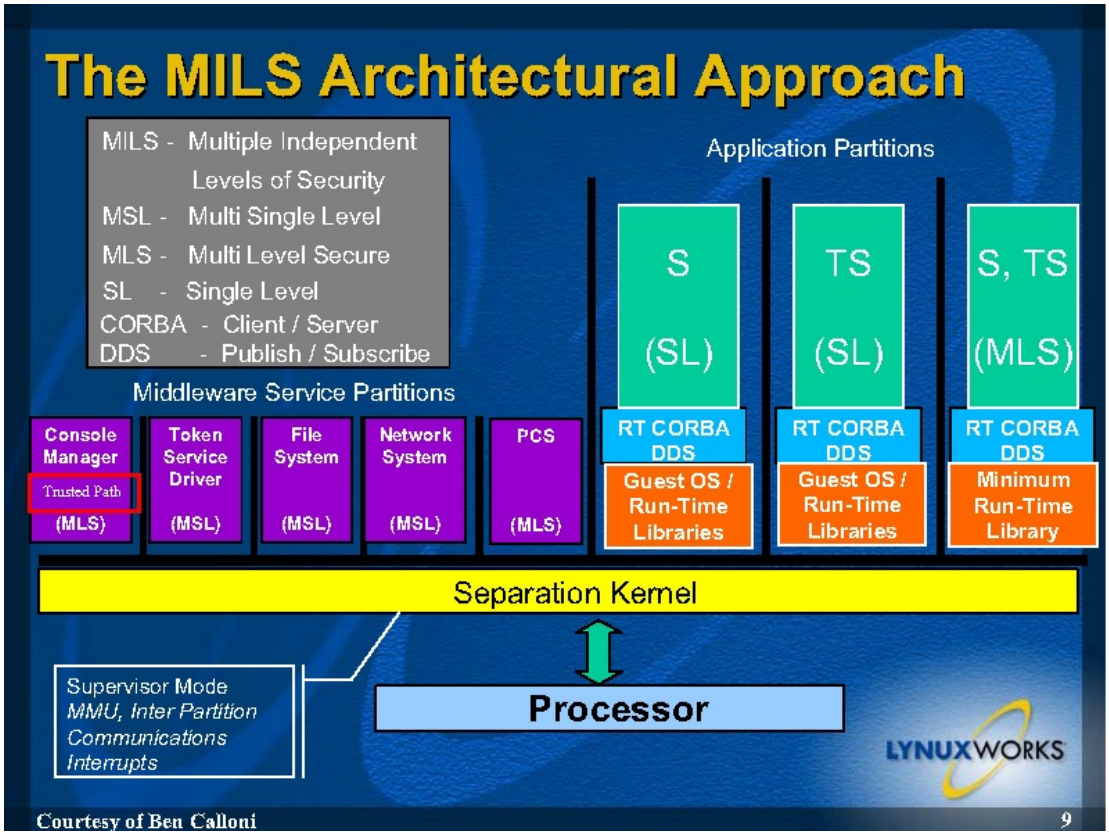


Figure 3.3 MILS workstation

4 Security evaluation

To be approved for handling of classified information, a system may need to be evaluated according to the Common Criteria (CC). For a complex information system this is an expensive and time consuming task. The effort required depends on the level of assurance that the system is aimed for.

4.1 The evaluation process

The Common Criteria (CC) [3] mandates what kind of documentations are required as input to an evaluation, and what has to be produced by the evaluation team before a security certificate for a product can be issued by the national security authority. One of the basic documents is the protection profile (PP). To ease the process of documentation, the PP can be standardized for a class of products. For the MILS Separation Kernel there exists a CC approved protection profile (SKPP) that can be used by all providers of this type of product.

Many parties participate in the evaluation process. The owner of the product has to plan for a certain assurance level. Then do the design, implementation and documentation according to this goal, and initiate the evaluation process, often supported by external security consultants.
The first part of the security evaluation is performed by an independent certified evaluation

company, and they produce an evaluation report. The final testing and approval of the evaluated product are done by the national security authorities. Only certificates of EAL4 products and below are mutually recognized between nations.

For high assurance systems (EAL6 and EAL7) the use of formal methods are mandated. For the highest level (i.e., EAL7) this includes the use of mathematical proof. This sets a practical limit for the size of a high assurance system, by existing methods said to be 2-3k lines of code. Traditional monolithic operating systems have never been certified higher than EAL5. This fundamental problem was the reason for John Rushby to propose the concept of separation in 1981, which later became the inspiration for the MILS architecture.

4.2 Compositional evaluation

Implementing a secure system on a MILS platform opens for splitting up the problem into smaller manageable modules that can be evaluated separately. This compositional evaluation as proposed by John Rushby and Rance DeLong [11;12], is based on a two-level approach to the system design and evaluation, i.e., the policy level and the resource sharing level. As mentioned earlier, the MILS separation kernel and the different partitions perform their local sub policies for each of the components. For the assurance of the local policy of a component, it should be as simple as possible. In fact, this should guide the design of a MILS system and how it is partitioned. For the complete system there has to be a policy architecture that integrates the different sub policies such that the policy for the complete system is met.

In a MILS system the sharing of resources is a different problem than policy, and therefore can be handled separately. The resource sharing level deals with the implementation of components, the information flow between them and how resources can be shared in the system in a way that do not compromise the policy of the system.

The methodology on how to perform a compositional design and evaluation are not yet available, but there is ongoing work at SRI, to provide a MILS Integration Protection Profile (MIPP). Such a formal integration methodology would be of great significance to reduce the certification costs of future MILS based products.

5 MILS Progress

The goal of the original MILS program led by AFRL was to fund evaluation and validation of MILS components, including the development of standard protection profiles (PP). The vision was to have a family of high assurance COTS products from different vendors that could be put together into complete systems of various kinds.

Since 2007 little information of the AFRL program have been published. We have been informed

that the program is still running but it is no longer focused on sponsoring evaluation of MILS components. That means the driving force is left to the industry where the work on developing MILS components is done, and the security evaluations have to be funded by the vendors themselves or by funding they manage to rise from other sources. More theoretical works are published by scientists as papers and presentations.

5.1 MILS vendors/products

This section mentions the major suppliers of MILS components, and in what segment they are active. This will change over time, so recent information about MILS components should be searched at the web sites of the vendors.

5.1.1 Green Hills

As the first MILS separation kernel product INTEGRITY-178B from Green Hills Software Inc [13] received its Common Criteria EAL6+ certificate September 1st 2008. As a follow up of this Green Hills has formed a new company, INTEGRITY Global Security, LLC. The charter of this new company is to use INTEGRITY as the foundation to protect government and corporate cyber assets, including safe Internet browsing, securing commerce transactions and protecting critical resources and infrastructure.

This first EAL6+ certificate is for a Compact PCI card with PowerPC. However, the formal methods and evaluation of the separation kernel was carried out on the code towards a hardware abstraction layer representing the actual hardware, i.e., the CPU and the chipset. A set of assumptions were made for this layer, and part of the evaluation was to verify that the target hardware met these assumptions. To port the INTEGRITY separation kernel to another hardware platform, the task will be to verify that the new hardware meets the same assumptions that were set for the original evaluation.

Green Hills has for some time had its MILS workstation, the INTEGRITY PC which is virtualization software running above INTEGRITY real time operating system with DO-178B Level A certification. INTEGRITY PC supports Linux and Windows as guest OSs.

5.1.2 LynuxWorks

In November 2008 LynuxWorks [14] announced their LynxSecure 2.0. They call it a next generation of separation kernel and hypervisor for high assurance systems. I.E. virtualization software for guest OS is part of the product. LynxSecure is designed and implemented from scratch to be able to reach an assurance level up to CC EAL7. They make use of automated formal methods, which is said to significantly reduce the time (and cost) to do the CC evaluation and subsequent re-evaluation. A specific evaluation will typical take place in cooperation with a customer on a selected target. Plans for a general purpose MILS workstation is not known, but they have sketches of such in their more general MILS presentations.

5.1.3 Wind River

In June 2008 Wind River [15] announced that its separation kernel, VxWorks MILS 2 is in evaluation to CC EAL6+. Wind River has been one of the major vendors for partitioning real time operating systems for the avionics with their VxWorks 653, and the MILS version leverages technology from this. The company expects their product to be used by those building multi level secure systems. The timescale is not known or if they have decided on a specific platform yet. Probably not, because on availability they state that "WxWorks MILS 2 certification timelines depend on customer-specific "Target of Evaluation", which are the actual targets (hardware) to be evaluated in a security analyses."

Wind River has also announced a trusted network stack product (IPSecure), and do work on the protection profile for MILS Network Subsystem (MNSPP).

5.1.4 Objective Interface

Objective Interface (OI) [16] is the major MILS middleware company. They had the idea of the PCS module as part of the architecture to create a distributed MILS, and they have drafted the protection profile for this (PCSPP). They have described a PCS-product (PCSexpress), but it is not commercially available yet. To call it a MILS product they need to fulfil the PCSPP, and have PCSexpress certified. OI also provide MILS adapted versions of more traditional middleware products: a real time version of CORBA (ORBexpress) and DSS (DSSexpress).

5.1.5 SYSGO

SYSGO [17] is a European company providing products and services in the area of RTOS and Network communication. They are aiming EAL5+ certification for their PikeOS virtualization platform. They state that PikeOS satisfy safety critical requirements like for avionics, and security requirements such as MILS (is MILS compliant). It is available for PowerPC, x86 and MIPS.

5.1.6 Other MILS actors

University of Idaho has been one of the major MILS actors from the academia. They were designated by the NSA as one of the US Centers of Excellence in Information Assurance in 1999, and have since then been a leader in the development of formal methods for software assurance. They also assist MILS vendors in their use of formal methods for products to be evaluated.

Another actor within MILS related research is SRI International, an independent non-profit research institute. Their Computer Science Laboratory led by John Rushby has research activities on formal methods and evaluation, and they also assist the industri.

Experts in security evaluation and formal methods are used as consultants by the developers of MILS components to prepare their products for evaluation. The security evaluation itself has to be set out to an independent company that is certified to do CC evaluation.

MILS industry partner also do research and test bed activities that contribute to the evolution of

the MILS architecture and concepts for utilizing it in future systems having safety and/or security requirements. This range from embedded systems for avionics, software defined radio and medical equipment to more general purpose secure military information systems. In particular will cross domain solutions including non military cooperation drive the requirements for high assurance components.

6 Applications of MILS

MILS' ability to provide strong separation between different partitions, in combination with the possibility to allow communication between selected partitions, makes it a suitable platform for a range of high assurance applications. In this section, we provide several examples of such applications where MILS may be utilized in order to realize high assurance systems.

6.1 Basic Usage

In this section we illustrate the basic usage scenarios of MILS.

6.1.1 Multiple OS Instances at Different Security Classifications

A common usage scenario of MILS is shown in Figure 6.1. In this scenario, two operating system (OS) instances, at different classification levels, are running within the same PC. Each operating system instance runs within a separate partition, potentially connected to a network at the corresponding classification level. By allowing both security levels to run on the same machine, sharing the same keyboard, mouse, and monitor, space, weight, and power can be saved. Apart from these benefits, however, such a solution fundamentally offers no additional functionality apart from what is available using two PCs. In fact, this seems to be true for most MILS applications, that the same could also be achieved using separate hardware. Nevertheless, the possibility to realize such solutions only using a single machine may be the deciding factor making the solutions practical.

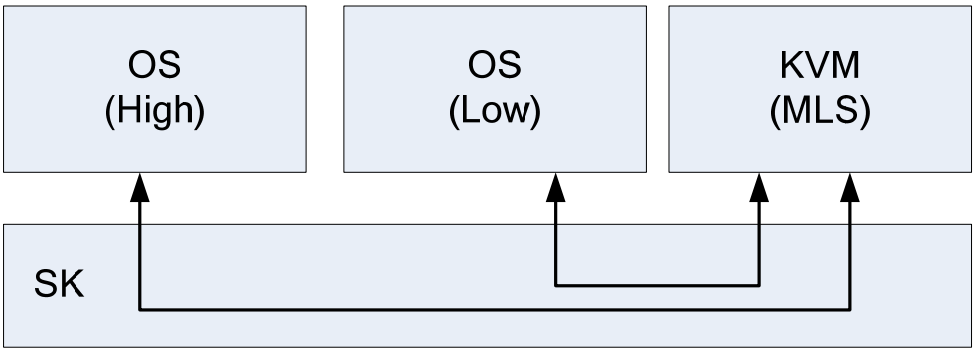


Figure 6.1 Multiple operating systems at different classification levels on the same PC. There is also an MLS partition for secure sharing of the keyboard, video, and mouse (KVM).

A typical addition to a system such as the one in Figure 6.1 would be the possibility to manually

cut-and-paste between the different OS partitions, either only from low to high or both ways. Obviously, the cut and paste-mechanism would also be subject to MLS requirements, and would typically be implemented in a separate partition connected to all the partitions for which the inter-partition cut-and-paste functionality should be enabled (similar to the KVM partition in the figure).

It should be noticed that the potential usage of separate partitions (for OS instances) is not limited to handling information at different classification levels on the same PC. The same functionality may for instance also be used in order to support the principle of least-privilege, where a user may switch between the different OS instances depending on the role he/she is acting in. With such a system, the role(s) of the user might also determine which partitions he/she would be able to access. An advantage of hosting a guest operating system within each partition is that it enables legacy/commodity applications to be used.

6.1.2 Providing Protection for Security Critical Components and Services

Another potential application of MILS is for providing protection of security critical components and services, as shown in Figure 6.2. In this way, the security critical service and its data are protected from being compromised by the remaining system. Such services may for instance include a signature service, where the signature application and the associated key(s) may be protected within a separate partition.

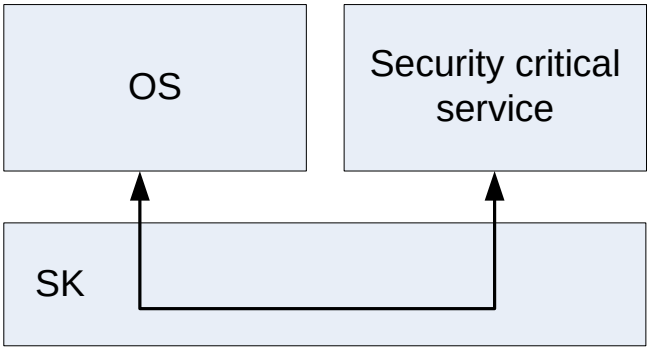


Figure 6.2 The security critical service and its data are protected within a separate partition.

In particular, such an approach is suitable in combination with a service oriented architecture, where service orientation provides for the decomposition of the system into services with well defined interfaces. By implementing the security critical functionality as a simple service with a minimal interface, and protecting the service within a separate partition, a high assurance implementation is facilitated. As an additional advantage, such an approach eases the reuse of security evaluated services/components in other systems.

Returning to our example in Figure 6.2, it may be noticed that a compromised OS partition could potentially submit falsified data to be signed. Thus, in order prevent that from succeeding, a signature service might also include a manual review of the information to be signed. Such a manual review might either be implemented within the same partition as the signature service, or

in a separate partition on the data path between the operating system and the OS. This leads us to another application of MILS.

6.1.3 Ensuring the Path of Information Flow (Non-Bypassability)

As mentioned previously, MILS provides for specifying which partitions are to be allowed to communicate, and in which direction(s) information may be allowed to flow. An implication of this is that MILS provides a high assurance basis for ensuring that security mechanisms can not be bypassed when information is exchanged between partitions. This is illustrated in Figure 6.3, where all data from partition A to partition B pass through the security mechanism in the middle partition. In this example, MILS also ensures that no information is allowed to flow in the opposite direction. Although the implementation of the security mechanism itself is still critical to the security of such a system, the enforcement of non-bypassability of the security mechanism is much simplified. Also, the strong separation between partitions protects the security mechanism from being compromised.

Examples of security mechanisms for which this can be utilized include encryption, access control, guards, auditing/logging, manual review (downgrader), security labelling, and so on.

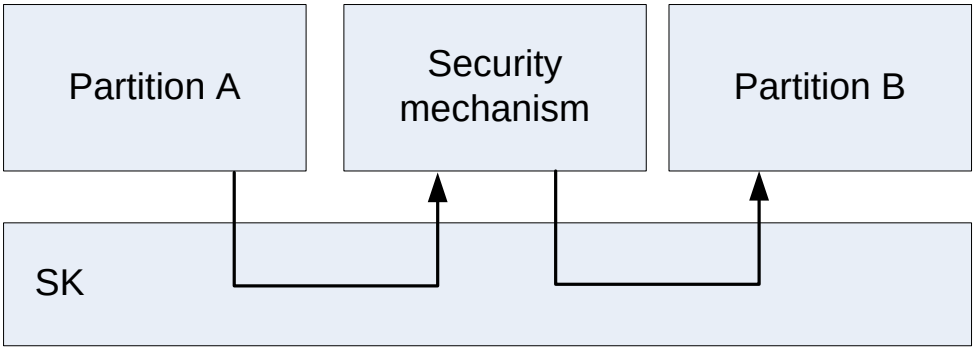


Figure 6.3 All data from partition A to partition B have to pass through the security mechanism. Furthermore, no information is allowed to flow in the opposite direction.

6.2 Compound Solutions

In this section the flexibility of MILS is illustrated by showing how MILS may potentially be used to solve some actual problems.

6.2.1 Two-Way Information Exchange between Security Domains/Levels

6.2.1.1 Background

In military systems, information is classified according to its sensitivity level, and only those with a clearance level equal to or above that of the information may be allowed to access the information. This results in systems with information flow control according to the Bell-LaPadula model. Following this model, information is only allowed to flow from a given security level to

an equal or higher security level (i.e., from low to low or from low to high, but not from high to low). A common way to satisfy this requirement is to have separate computer systems for each classification level, where communication between each system is non-existent, or at least very limited, due to physical separation.

While physical separation is effective for enforcing information flow control, without imposing excessive assurance costs, it often also results in inflexible systems that unnecessarily prevent information sharing. Thus, there is often a strong trade-off between system assurance and system usability. Manual review and declassification/release has been the most common solution to this usability problem. With manual declassification/release, information to be passed from high to low is reviewed by a human and then released manually. With the introduction of a network based defence, however, both the amount of information and the importance of having the most up-to-date information will likely increase. Thus, manual review and declassification alone does not appear to provide a suitable solution for use in a network based defence. In particular, the communication in a network based defence will to a larger extent consist of machine-to-machine communication, where human review becomes both impractical and error prone. Thus, in a network based defence, more automated information flow control is needed.

The alternative to physical separation is to have information flow control enforced by the system itself. Such systems capable of securely handling information and users at different security levels are referred to as multilevel secure (MLS) systems. Although this may seem like an ideal solution, the costs of assurance in MLS systems are prohibitive for many systems. Furthermore, the highest assurance levels are in practice only achievable for systems with limited functionality/complexity.

In view of the disadvantages of both systems based on physical separation and MLS systems, a hybrid approach is often desirable. Such hybrid approaches have traditionally been achieved to some extent by maintaining a high degree of physical separation between systems at different security levels, but facilitating a restricted information flow. For instance, the security policy may allow information to flow from low to high, e.g., from a confidential system to a secret system.

Such a security policy can be enforced using a diode or network pump (only allowing information to be sent in one direction). A network pump [18] differs from a diode by allowing acknowledgements to be sent in the reverse direction. In order to limit the covert channels that would result from having acknowledgments flow from high to low, the network pump buffers the packets from the sender (low) while sending the packets to the receiver (high). The acknowledgements from the receiver are received by the network pump, which again sends the acknowledgements to the sender (low) with randomized timing. This way, the covert channel bandwidth due to acknowledgements from high to low is significantly reduced.

Such a scheme does not completely remove covert channels from high to low though. For instance, the acceptance or rejection of a connection by the receiver provides a covert channel which can be used to communicate one bit per connection [19]. Thus, we see that the

reliable/acknowledged delivery provided by the network pump comes at the cost of an increased security risk.

Although there are systems where such an increased security risk in the form of covert channels is not acceptable, there are also many systems where the limited risk of such a covert channel is outweighed by its benefits. Furthermore, by strictly restricting the computers/applications allowed to communicate through a network pump, the risk of the covert channel can be further reduced.

As may be observed, however, a system with a network pump still offers very limited flexibility. For instance, there is no automatic way to pass information from a high system to a low system although this information may be classified at or below the low level. Likewise, there is no way for a system at the high level to submit a request (e.g., a request for information) to a system at the low level.

6.2.1.2 A MILS Based Solution

Assuming that all information objects are correctly labelled according to their classification level, it is possible to have a guard positioned between the security domains to perform information flow control based on the security labels. The correctness of such a scheme relies on two assumptions, that is, correct guard behaviour and correct labelling of information.

The guard has only limited functionality and can be strongly separated from other components (e.g., by residing within a separate partition). This strong separation and limited complexity facilitates a high assurance guard implementation.

The correctness of the security labels on the other hand depends on two factors:

- That the security label of an information object can not be illegally changed (i.e., from a higher to a lower security classification), either by modifying the security label (or the information object itself) or by replacing the security label (integrity).
- Correct labelling of the information objects in the first place (assurance).

Starting with the first requirement, the security labels can be attached to the data to which they apply using a strong cryptographic binding, such as a digital signature, ensuring the integrity of both the security label and the data. With this in mind, we will now consider how MILS can be used to provide assurance that information objects are in fact labelled correctly in the first place (i.e., the second requirement).

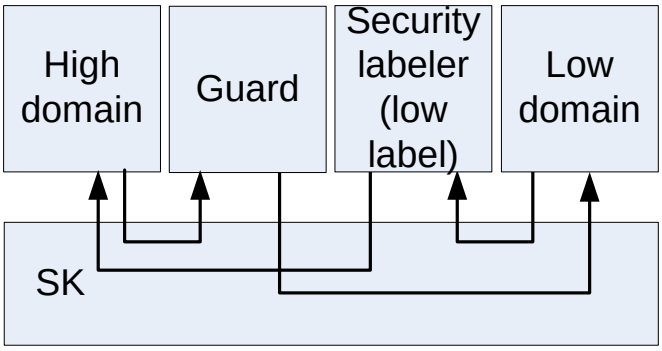


Figure 6.4 A combined guard and security labeller solution.

Consider the MILS based architecture shown in Figure 6.4. In this scenario, we have a guard preventing information classified at the high level from being sent to the low domain. As opposed to a one-way diode or network pump, however, the guard allows information with a valid low security label to be passed from the high to the low domain. Thus, data may be sent from the low domain to the high domain, and then from the high domain and back down to the low domain as long as the data remains unchanged with a valid security label according to the low level. We do not explicitly label high information in this example scenario, as any information without a valid low label is blocked by the guard.

Let us take a closer look at this architecture. In particular, it may be noticed that the effectiveness of the guard is tightly dependent on the security labeller attaching/signing the low security labels. Thus, although the low domain might be subject to a less restrictive security policy than the high domain, the security labeller must be subject to the same high level of assurance as the guard. In the case that the cryptographic key used for attaching the security labels were compromised, and somehow ended up within the high domain, the key could be used to leak information from the high to the low domain by attaching false security labels. For this reason, the key used for signing the security labels is only available within the security labeller partition.[1]

The public/private key pair used for binding the security labels may be strongly bound to the security level of the security label. Thus, a key pair used in a high security domain for binding a high security label would not be valid for binding a security label specifying a classification below the high level.

Furthermore, due to the MILS configuration, only data within the low domain can be sent to the low security labeller, and hence have a low security label attached. Thus, in order for a low security label to erroneously be attached to some data originating at the high level, the data would first need to leak from the high to the low level (which is prevented by the guard).

It may also be noted that numerous variations of the above scheme exists. For instance, a domain might be connected to several other domains through such guards/labellers, in which case

[1] Notice that it is also possible to use symmetric cryptography with such a scheme. In that case, the same key would be available to both the security labeler and the guard.

information from a low domain entering a higher domain may be passed through to another low domain. The solution can also be extended to label and filter information in both directions. Furthermore, the solution may be extended with separate partitions for manual review and release as well as auditing/logging partitions. By including a network interface in the end-partitions, the guard/labeller can also be used between networks. In fact, an advantage with such a solution is that the security critical components (i.e., the guard and labeller) can be reused for various applications.

Another possibility may also be to combine a variation of the presented scheme with a high-water mark policy, where a partition is initialized at the lowest classification level, and where the guard/labeller increases the security classification of the partition depending on the classification of the information entering the partition.

In actual scenarios it may also be required to perform security screening (e.g., virus scanning) of the data being sent between security domains. Such additional security screening mechanisms can be included in separate partitions on the communication path between the respective partitions.

6.2.2 Multiple Service Instances (and Reduction of Data Replication)

In this section we use the guard/labeller from the previous section, in order to provide multiple instances of the same service at different classification levels while at the same time reducing data replication. As shown in Figure 6.5, the service instance in the low partition stores its data in a high partition storage. By having a low security label attached to the data before storage, the data can be retrieved through the guard at a later time (effectively providing MLS storage).

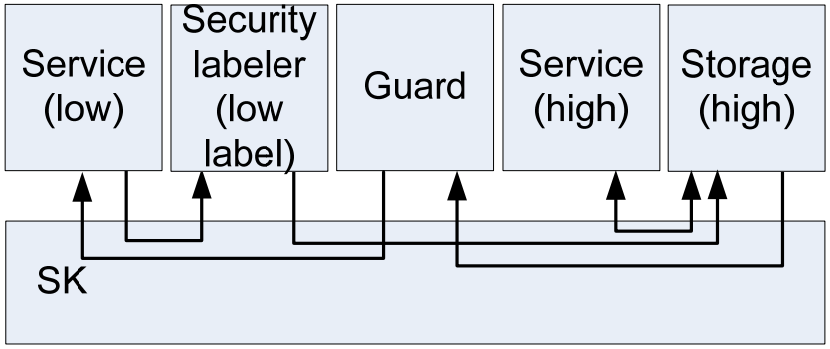


Figure 6.5 Two instances of the same services relying on the same data storage.

When a client attempts to access this service, it is given access to the service corresponding to its security clearance. The low service will return a reply based only on the information classified at the low level, while the high service will return a reply based on all the available data. A limitation with this approach though is that all replies from the high service is classified as high even if solely based on data classified at the low level. In order to avoid this limitation, the high service (or alternatively the high client) would need to be allowed to invoke the low service.

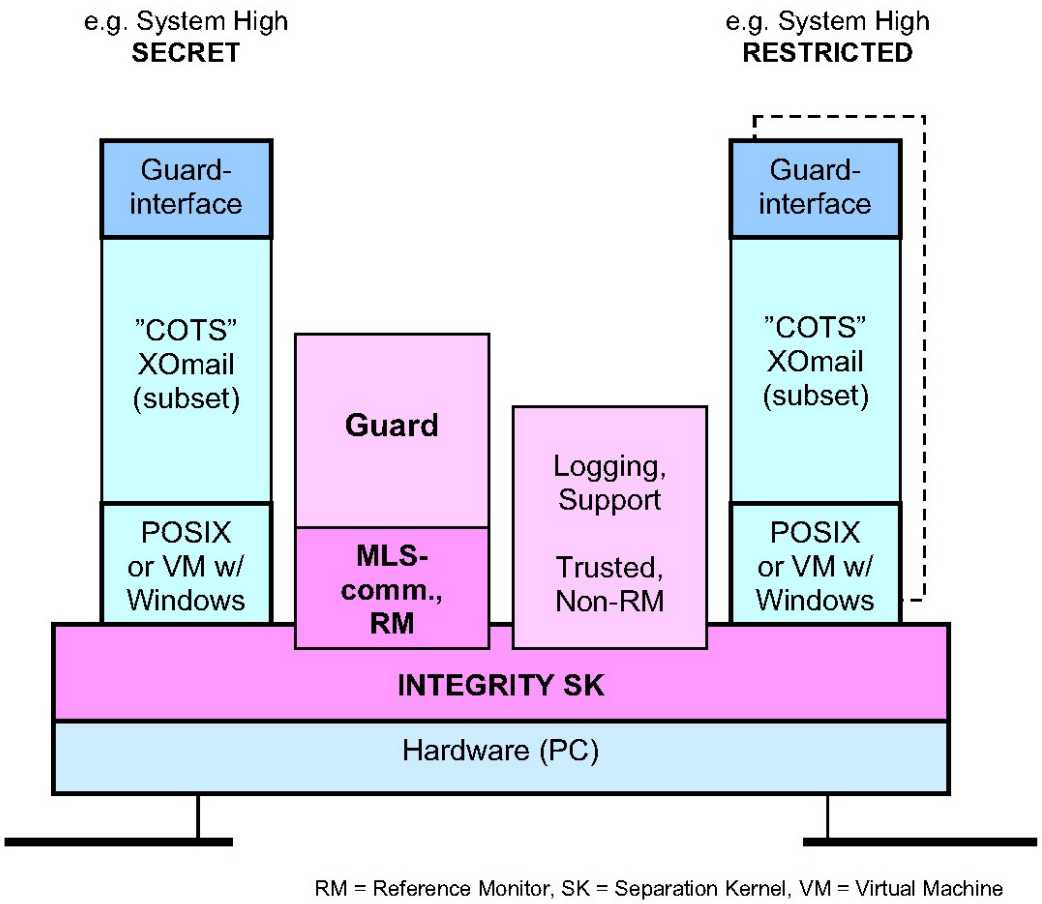
6.3 Short term applications of MILS

6.3.1 Military message security guard for the Norwegian defence

The Norwegian defence has had an MLS military message handling system (MMHS) in operation since ca 1990. It conforms to STANAG 4406 and was certified according to TCSec/Orange Book, level B2. Now the certificate has expired, the hardware of the certified version is obsolete and the UNIX operating system, AT&T System V MLS is not maintained any more.

The system, XOMail by THALES Norway, has also been delivered on other platforms, like Solaris and Windows, but these have not been certified. In the plans for updating the system, THALES did a study for the Norwegian defence. There is still an MLS requirement, but now limited to the message guard function, to be able to interconnect different security domains/nations.

The recommendation from this study is to use a MILS separation kernel as platform for the message guard for the next generation of MMHS for the Norwegian defence, see Figure 6.6.



The figure is taken from a study report by THALES Norway

Figure 6.6 MILS based military message guard.

6.3.2 Army tactical equipment

For man pack equipment in field size, weight and power consumption are critical factors. The soldier often ends up carrying separate units for communication and information handling. If he needs to access different systems and domains, existing solutions are to have separate systems, which means more to carry. With MILS based systems it may be possible to integrate different information systems on the same physical platform, even if they are of different classification. By utilizing the virtualization technique provided for MILS separation kernels, this can be done by existing applications without modification. Also encryption, i.e., IPsec, can be integrated on the same platform.

Military vehicles also have limited space. In particular will the introduction of additional hardware be problematic. In that sense a MILS platform will be far more flexible and open for integration of new system without having to introduce yet another box.

Some software radio vendors make use of MILS separation kernels in their systems. A likely evolution would be that future software radios are prepared for adding the customers' applications in "free partitions" on the radio platforms.

7 Conclusions

Implementation of the NNEC vision to open for timely and cross-domain information flow with high degree of mobility will imply strong assurance requirements not met by existing solutions. The MILS architecture and its related products are strong candidates as platforms that have a potential to meet the requirements set for high assurance systems.

The application of this technology is new when it comes to secure systems, but the technology is well proven and mature for embedded systems with safety requirements. Since these two categories have great similarities, it is believed that the MILS architecture will survive as a solution for high assurance systems. This expectation is supported by the fact that one of the MILS separation kernels was certified to EAL6+ recently.

Although there have been serious attempts to bring security into more general operating systems, like for multi level secure (MLS) versions of UNIX, these systems end up being much more complex with a large security kernel. As a consequence, they are expensive to evaluate and maintain and will typically be lagging behind in the technological evolution. Furthermore, it is not possible to evaluate such monolithic systems to the level required for high assurance systems.

An enabler for the introduction of MILS type of products is that an evolutionary approach is possible. Existing applications can run as virtual machines on MILS platforms to have some short term gains. The long term goal for secure systems and security components can be implemented step by step.

Still there is a lot of work to be done by the MILS community within research and development that will require sponsors and investments. However, the evolutionary approach will open for this since the return of investments can be expected to come rather quickly.

In this report we have provided an overview of MILS and its applications. In our continued work we will further investigate the use of MILS for specific applications and scenarios. Through this continued work we plan to gain first hand experience with the use of MILS, enabling us to better evaluate its suitability as a platform for future systems.

Bibliography

[1] J. Alves-Foss, P. W. Oman, C. Taylor, and W. S. Harrison, "The MILS architecture for high-assurance embedded systems," *International Journal of Embedded Systems*, vol. 2, no. 3, pp. 239-247, 2006.

[2] W. M. Vanfleet, I. A. Luke, R. W. Beckwith, C. Taylor, B. Calloni, and G. Uchenick, "MILS: Architecture for High-Assurance Embedded Computing," *Crosstalk: The Journal of Defense Software Engineering*, 2005.

[3] Common Criteria," http://www.commoncriteriaportal.org.

[4] W. S. Harrison, N. Hanebutte, P. Oman, and J. Alves-Foss, "The MILS Architecture for a Secure Global Information Grid," *Crosstalk: The Journal of Defense Software Engineering*, vol. 18, no. 10, 2005.

[5] NIAP-CCEVS," http://www.niap-ccevs.org.

[6] J. M. Rushby, "Design and verification of secure systems," *ACM SIGOPS Operating Systems Review*, vol. 15, no. 5, pp. 12-21, 1981.

[7] R. J. DeLong, "MILS: Protecting our most vital systems," *Military Embedded Systems*, 2007.

[8] B. Randell and J. Rushby, "Distributed Secure Systems: Then and Now," *Computer Security Applications Conference*, 2007, pp. 177-199.

[9] G. M. Uchenick, "Partitioning Communciations System for safe and secure distributed systems," *Digital Avionics Systems Conference*, 2007.

[10] D. Kleidermacher and M. Wolf, "MILS virtualization for Integrated Modular Avionics," *Digital Avionics Systems Conference*, 2008.

[11] C. Boettcher, R. DeLong, J. Rushby, and W. Sifre, "The MILS component integration approach to secure information sharing," *Digital Avionics Systems Conference*, 2008, pp. 1-12.

[12] Rushby, J. and DeLong, R., "Compositional Security Evaluation: The MILS approach," http://www.csl.sri.com/~rushby/slides/iccc07.pdf.

[13] Green Hills," http://www.ghs.com/.

[14] LynuxWorks," http://www.lynuxworks.com.

[15] Wind River," http//www.windriver.com.

[16] Objective Interface Systems," http://mils.ois.com.

[17] SYSGO," http://sysgo.com.

[18] M. H. Kang, I. S. Moskowitz, and D. C. Lee, "A network pump," *IEEE Transactions on Software Engineering*, vol. 22, no. 5, pp. 329-338, 1996.

[19] A. Aldini and M. Bernado, "Measuring the Covert Channel Bandwidth in the NRL Pump," http://www.di.unipi.it/~troina/mefisto/drafts/NRLPumpDraft.ps, 2004.