Applying MILS to multicore avionics systems

Eur Ing Paul Parkinson FIET
Principal Systems Architect, A&D

EuroMILS Workshop, Prague, 19th January 2016

WIND

AN INTEL COMPANY

© 2016 Wind River. All Rights Reserved.

Agenda

- A Brief History of MILS
- Implementation Considerations
- MILS & Multicore Convergence
- MILS Multicore Implementation on P4080
- MILS Multicore Use Cases
- Conclusions

Applying MILS to multicore avionics systems

Paul J. Parkinson
Principal Systems Architect
Wind River
Swindon, United Kingdom
Paul.Parkinson@windriver.com

***Abstract*— The implementation of the Multiple Independent Levels of Security (MILS) software architecture on modern microprocessor architectures has become technically feasible in recent years. This allows MILS-based systems to host applications and data of multiple security classifications concurrently on a uniprocessor platform at affordable cost. In this paper, the potential requirements for the implementation of a separation kernel to support MILS systems on multicore processor architectures will be considered, and the design challenges associated with its potential implementation on the NXP (formerly Freescale) QorIQ™ P4080 multicore processor will be discussed. Finally, the potential use of a MILS Multicore separation kernel in two use cases will be presented - a Cross-Domain System (CDS) network gateway, and a Multi-Level Secure (MLS) Integrated Modular Avionics (IMA) platform.**

Keywords—MILS; multicore; security; MLS; CDS; ADN

I. Adoption of Multiple Independent Levels of Security

Historically, commercial organizations and governments have categorized information at different security classifications, based on varying criteria including information value, sensitivity, and the impact of disclosure.

Information at different security classifications was traditionally physically isolated in separate domains. The methods used to enable authorized information flows between security domains have varied, but have often involved manual transformation of information which has fundamentally limited the speed of analysis of information and decision-making.

More recently, there has been a drive towards automation of the information flow process between different security domains. This enables decision-making to be accelerated, in order to provide benefits to applications as diverse as commercial business and banking operations, through to sharing information with coalition forces in theatre operations.

Initially, these multilevel secure computer systems were built using multiple, physically separated computers, networks, and displays. This technique, known as "air gap" security, required expensive equipment and occupied a large footprint in terms of Size, Weight and Power (SWaP). Whilst there have been efforts to address the multi-level security requirement through the development of monolithic, secure operating systems running on a single computing platform, their development and security certification would have taken ten or more years and at unaffordable cost due to the large size and complexity of the trusted computing base (TCB) [1].

In 1984, John Rushby proposed an alternative approach for secure embedded systems, utilizing a small trusted computing base as part of a layered software architecture, providing separation between different domains on a single processor [2]. This provided the foundation for the Multiple Independent Levels of Security (MILS) software architecture which was presented by Mark Vanfleet at the Open Group Security Forum in 2002. The MILS architecture provides a means of overcoming the development time and certification cost issues associated with large monolithic operating systems, through the use of a separation kernel (SK) built on four fundamental security policies:

- *Information Flow*. This defines the permitted information flows between partitions.
- *Data Isolation*. This ensures that a partition cannot access resources in other partitions.
- *Periods Processing*. This ensures that applications within partitions execute for the specified duration in the system schedule.
- *Fault Isolation*. This defines that a failure in one partition does not impact any other partition within the system.

Fig. 1. Multiple Independent Levels of Security (MILS) architecture

These four policies create an architecture where the separation kernel is *Non-Bypassable, Evaluatable, Always Invoked* and *Tamper Proof*, which is known as *NEAT*. This means that the size of the trusted computing base implemented by the separation kernel, shown as "RTOS Micro Kernel (MILS)" in Fig. 1, has the potential to be very small, as all

© Copyright 2016 Wind River. All rights reserved.

© 2016 Wind River. All Rights Reserved.

AN INTEL COMPANY WIND

Unintended information disclosure?

Downloaded from: http://www.flickr.com/photos/neilrickards/141523386/
Image © Neil Rickards. Creative Commons License

© 2016 Wind River. All Rights Reserved.

A Brief History of MILS


Design and Verification of Secure Systems

Reprint of a paper presented at the 8th ACM Symposium on Operating System Principles, Pacific Grove, California, 14–16 December 1981. (ACM Operating Systems Review Vol. 15 No. 5 pp. 12-21)

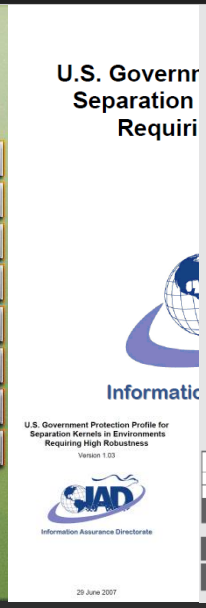
John Rushby*
Computer Science Laboratory
SRI International
Menlo Park CA 94025 USA

Abstract

This paper reviews some of the difficulties that arise in the verification of kernelised secure systems and suggests new techniques for their resolution.

It is proposed that secure systems should be conceived as distributed systems in which security is achieved partly through the physical separation of their individual components and partly through the mediation of trusted functions performed within some of those components. The purpose of a security kernel is simply to allow such a 'distributed' system to actually run within a single processor; policy enforcement is not the concern of a security kernel.

This approach decouples verification of components which perform trusted functions from verification of the security kernel. This latter task may be accomplished by a new 'verification technique called 'proof of separability' which explicitly addresses the security relevant aspects of interrupt handling and other issues ignored by present methods.

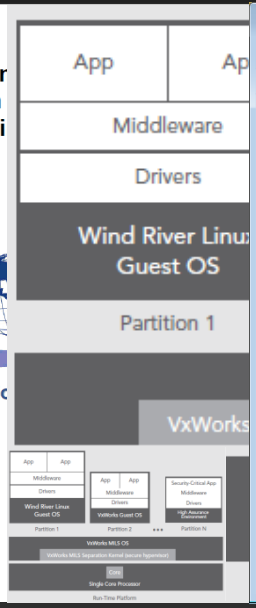

U.S. Governr
Separation
Requiri

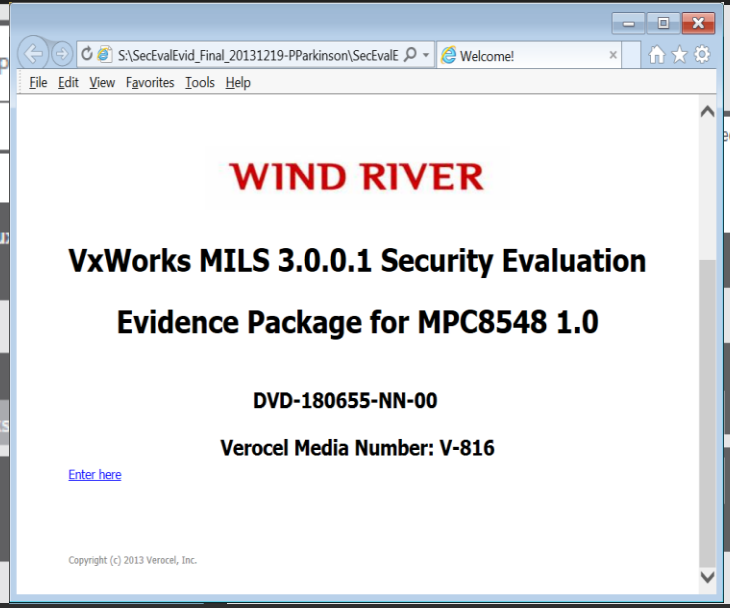
Informatic

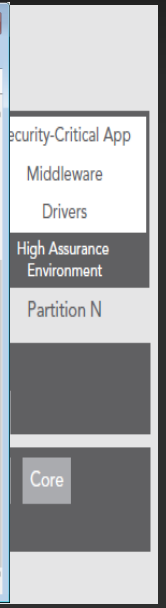
U.S. Government Protection Profile for Separation Kernels in Environments Requiring High Robustness
Version 1.03

Information Assurance Directorate

29 June 2007







John Rushby paper	W. Mark Vanfleet Open Group	Hardware virtualisation	SKPP v1.03	VxWorks MILS 2.0 (Single-core)	VxWorks MILS EAL6+ Package	VxWorks MILS 3.0 Multicore Edition
1981	2002	~2007	2007	2009	2013	2013

© 2016 Wind River. All Rights Reserved.

AN INTEL COMPANY

Separation Kernel Architecture: Four Fundamental Security Policies

Information Flow	Defines permitted information flows between partitions
Data Isolation	Ensures that a partition cannot access resources in other partitions
Periods Processing	Ensures that applications within partitions execute for the specified duration in the system schedule
Fault Isolation	Defines that a failure in one partition does not impact any other partition within the system

© 2016 Wind River. All Rights Reserved.

Implementation Considerations

Covert Channels of Communication

- Covert channels provide unintended channels of communication between isolated applications, enabling implicit communication of binary values
- Types of covert channel:
  - **Covert Timing Channel** can occur when there is variation in duration of partition's execution (partition jitter)
  - **Covert Storage Channel** includes *"all vehicles that would allow the direct or indirect writing of a storage location by one process and the direct or indirect reading of it by another"* [TCSEC]

© 2016 Wind River. All Rights Reserved.

Implementation Considerations

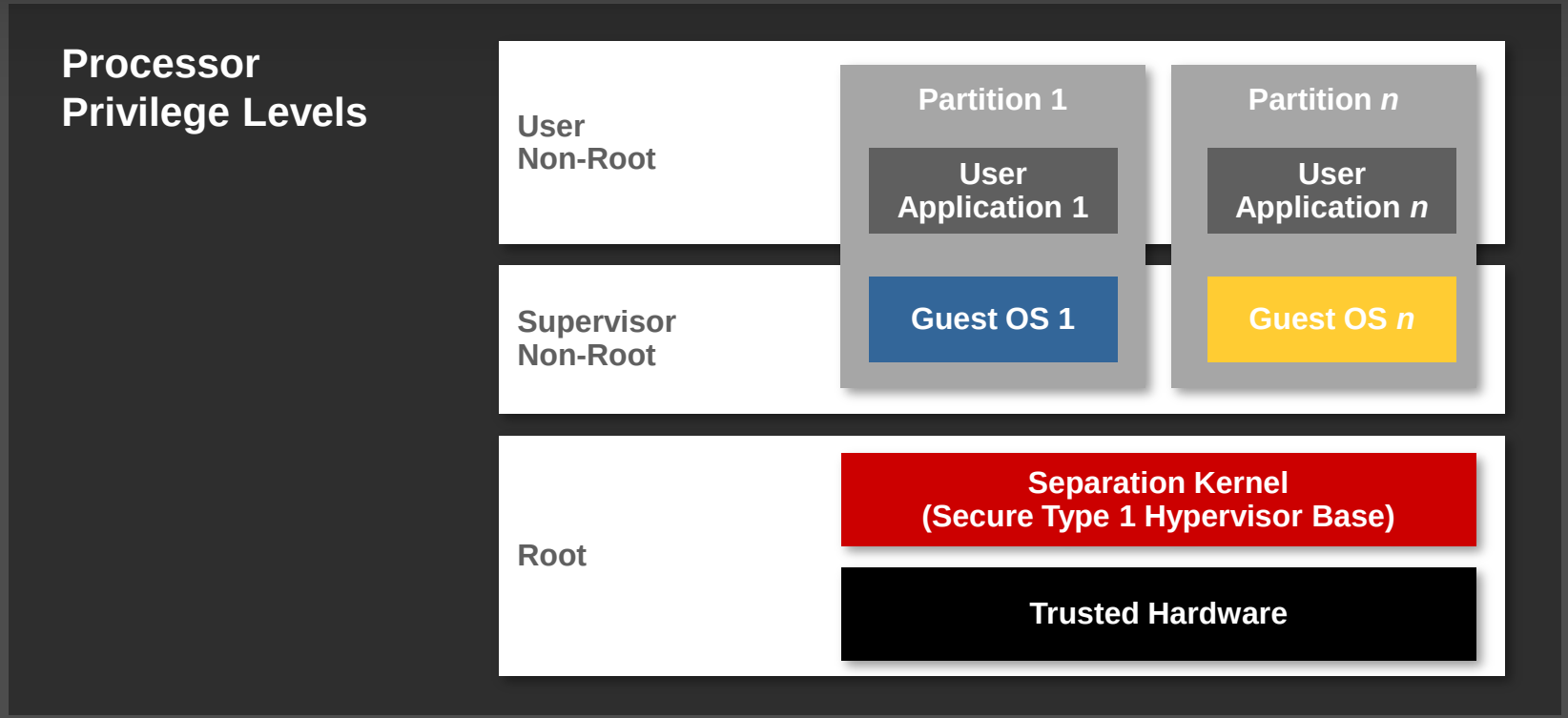
Virtualization Approaches

- Processor hardware virtualization support enables:
  - Hypervisor-based SK to run Guest OS in virtualized environment
  - Better enforcement of data isolation & data flows
  - Reduced potential of covert channels
- Two virtualization approaches often used:
  - **Paravirtualization** where guest OS is modified for common resource execution
  - **Full virtualization** where guest OS runs unmodified due to hardware virtualization support

Hypervisor must be provably secure
to withstand threats of high attack potential

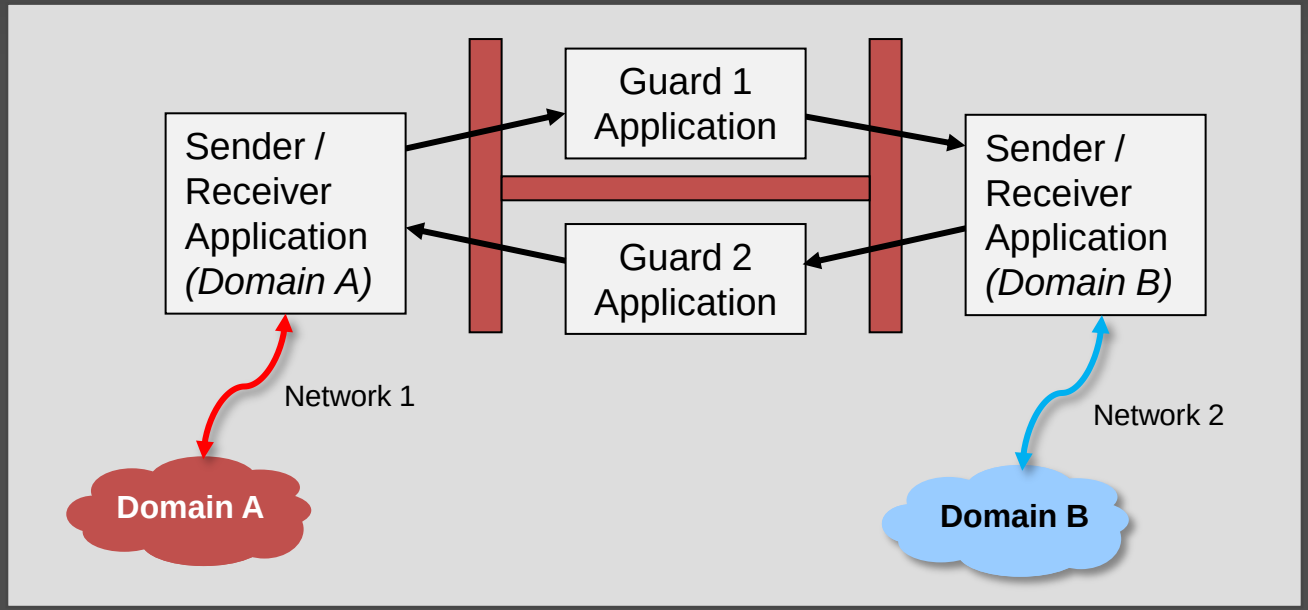
© 2016 Wind River. All Rights Reserved.

MILS Hypervisor Virtualisation Model



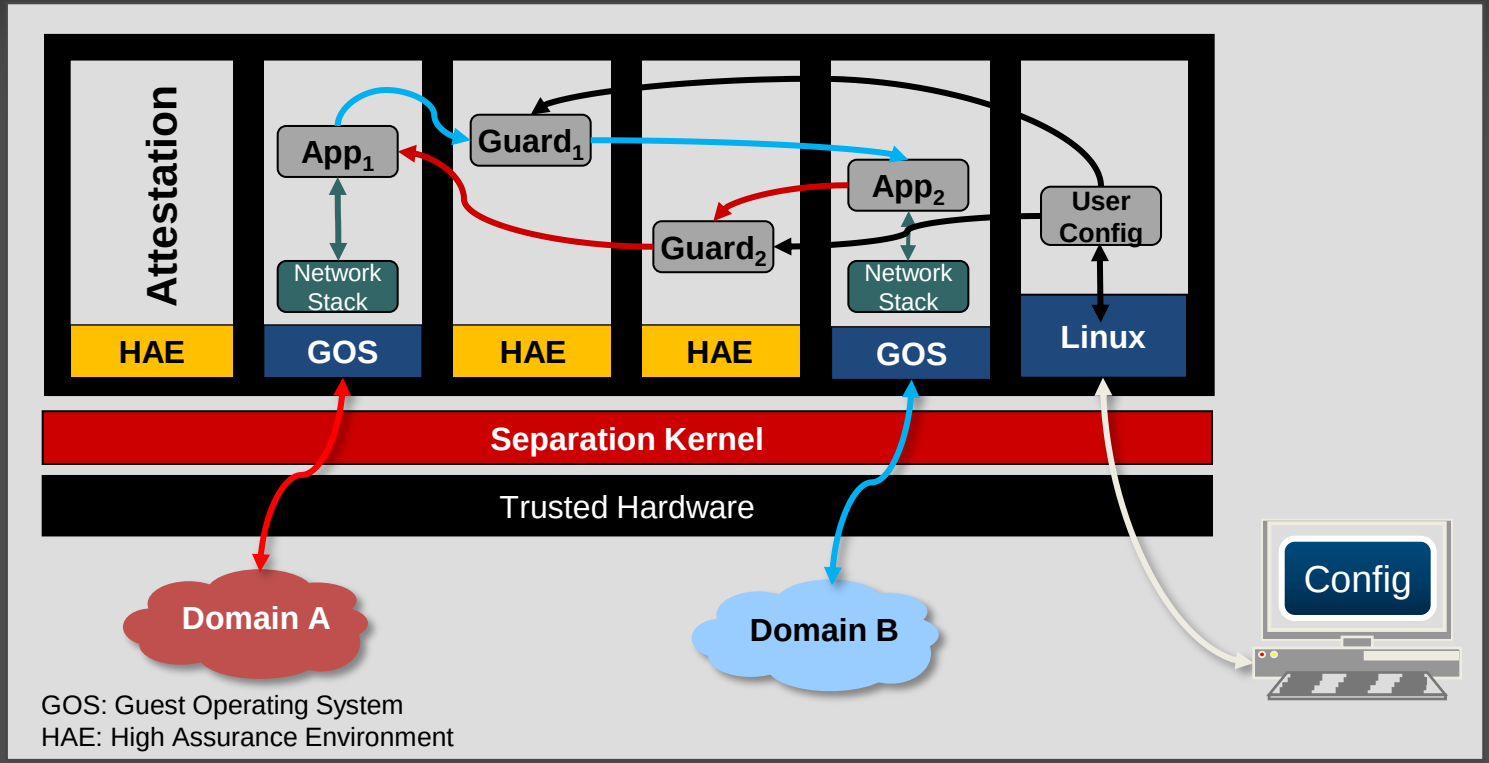
© 2016 Wind River. All Rights Reserved.

MILS Cross-Domain System use case



© 2016 Wind River. All Rights Reserved.
AN INTEL COMPANY
WIND

Notional MILS-based Cross-Domain System network gateway



© 2016 Wind River. All Rights Reserved.
AN INTEL COMPANY WIND

MILS & Multicore Convergence

- Need to exploit multicore performance in MILS systems
  - Enables consolidation of ‘air-gap’ security systems
- Primary multicore architectural challenges:
  - **Application isolation**
  - **Core separation**
  - **Resource sharing**
- FAA multicore safety research
  - MPC8572 multicore architecture
  - Found shared resource contention - determinism, denial of service
  - CAST-32 paper provides additional objectives for multicore safety
  - ARINC 653 Specification updated to support development of multicore IMA systems
- High-assurance security systems overriding concern:
  - **Covert channels** of communication with **increased bandwidth**

© 2016 Wind River. All Rights Reserved.

MILS Multicore Implementation on P4080

- Multicore Software Architecture
  - Many permutations possible on 8 core processor
  - AMP may provide better security characteristics than SMP
  - CoreNet 36bit (64GByte) address space per core, 32bit execution space per core
- Core Virtualisation
  - 3 privilege levels on P4080, enables use of Guest OS with memory protection
  - Full hardware virtualisation support reduces guest OS implementation effort
- Secure Boot
  - Initialisation of each core to known state and defined boot order
  - Automated tool support to provide continuous validation of system configuration to ensure security issues not introduced through boot of multiple cores in wrong order

© 2016 Wind River. All Rights Reserved.

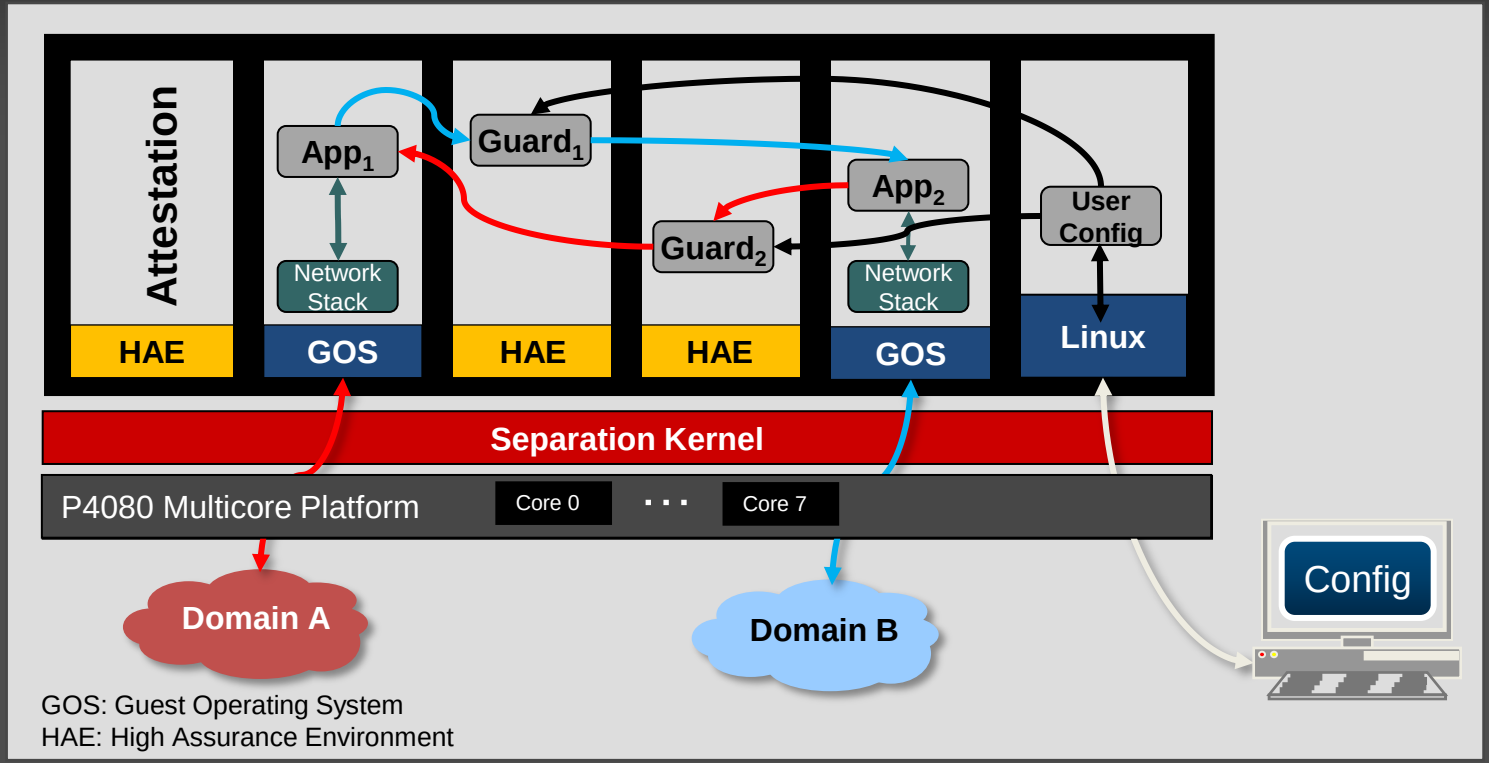
MILS Multicore Implementation on P4080 (2)

- Core Schedule Synchronisation
  - MILS unicore systems handle different security domains *sequentially*
  - MILS multicore has potential to handle different security domains *concurrently*
  - Potential for increased bandwidth of covert channels
    - Mitigation via P4080 core separation (memory ranges, L2 cache, 2 memory controllers)
    - Use of Synchronised Time-Sliced Scheduling model (example to follow)
- Inter-Core Communications
  - Needs to be transparent to application
  - Implementation using Secure IPC with ARINC 653 APEX port interface
- Independent Core Configuration
  - Need to support incremental composition of a system
  - Use of XML-based configuration for individual cores and platform with automated tool support and continuous validation

© 2016 Wind River. All Rights Reserved.

Notional MILS-based Cross-Domain System

Multicore Implementation Considerations

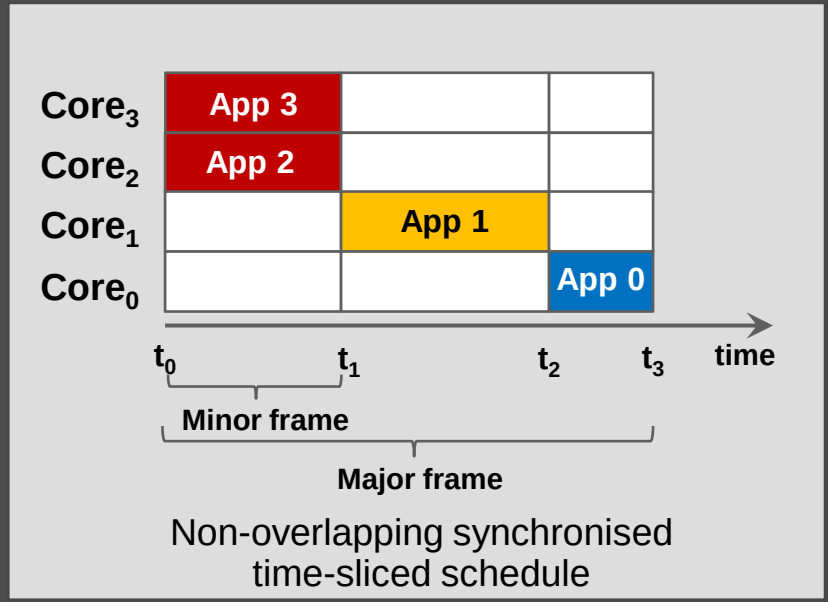


© 2016 Wind River. All Rights Reserved.

Notional MILS-based Cross-Domain System

Multicore Implementation Considerations

- Minimise potential for covert channels
- Allocation of applications to different cores
- Time-sliced synchronised scheduling
  - Limit number of security levels or domains processed at same time



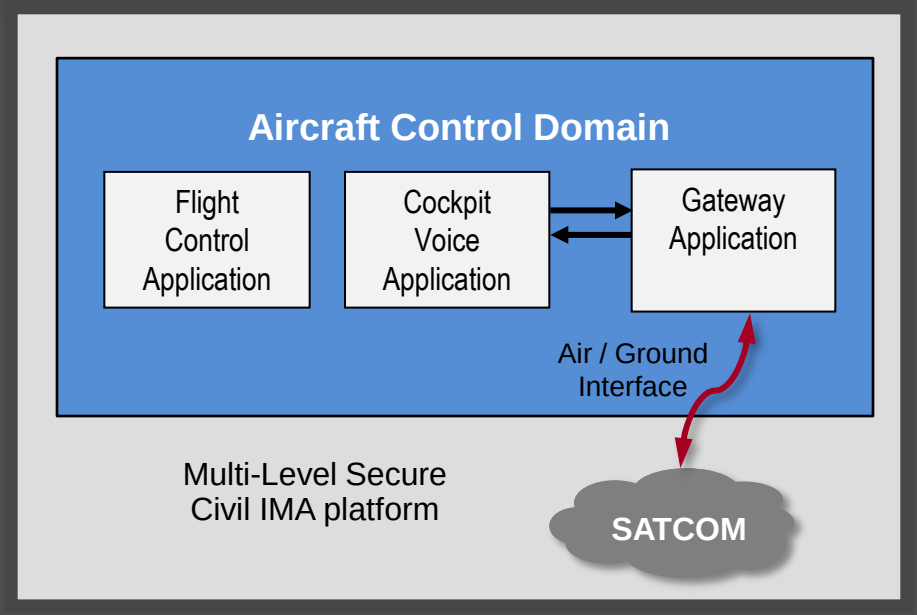
Non-overlapping synchronised time-sliced schedule

© 2016 Wind River. All Rights Reserved.

MILS Integrated Modular Avionics use cases

Multi-Level Secure IMA

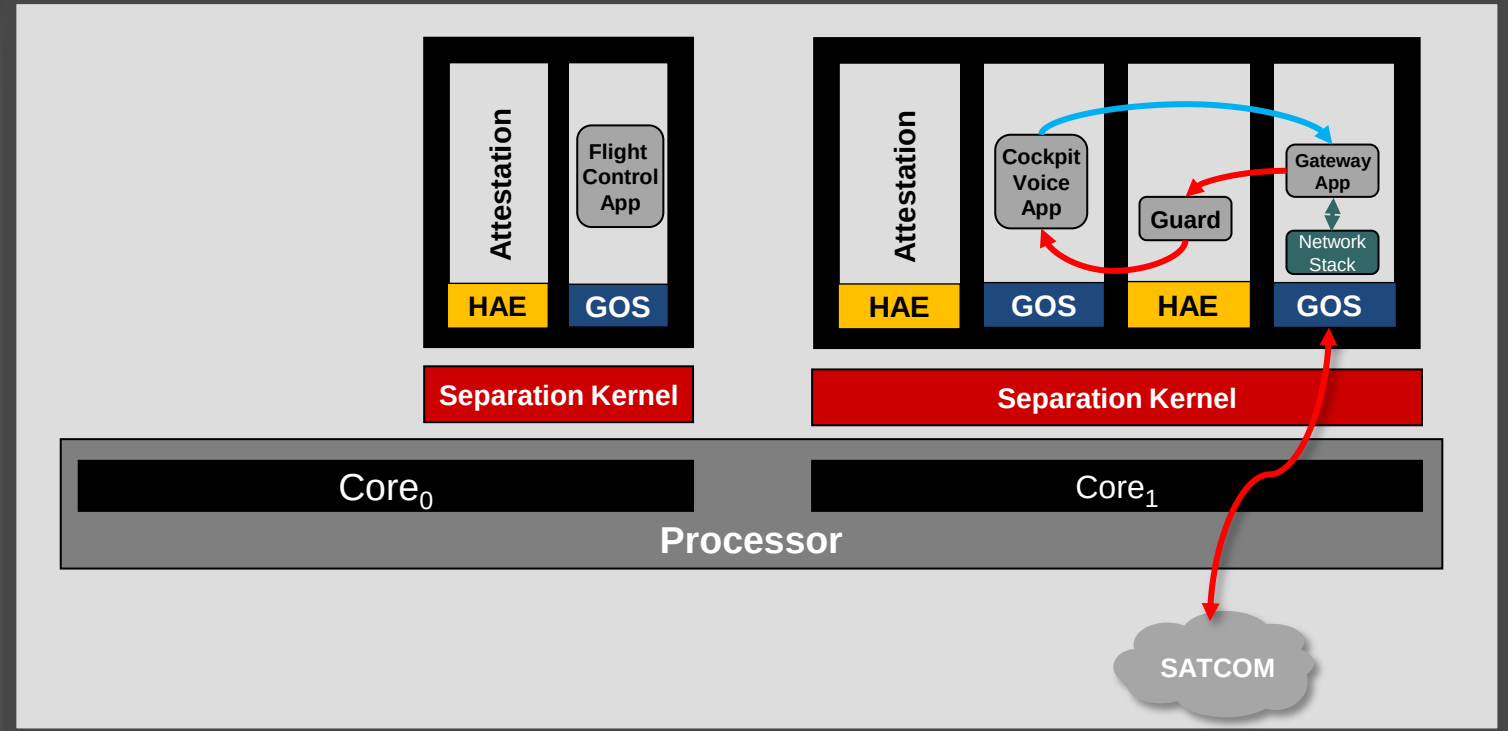
- IMA systems deployed using ARINC 653 architecture
- Growing requirements to support multi-level secure (MLS) in IMA environment
- Can be addressed using MILS:
  - Enforce information flows
  - Use of guards



© 2016 Wind River. All Rights Reserved.

MILS Integrated Modular Avionics use cases

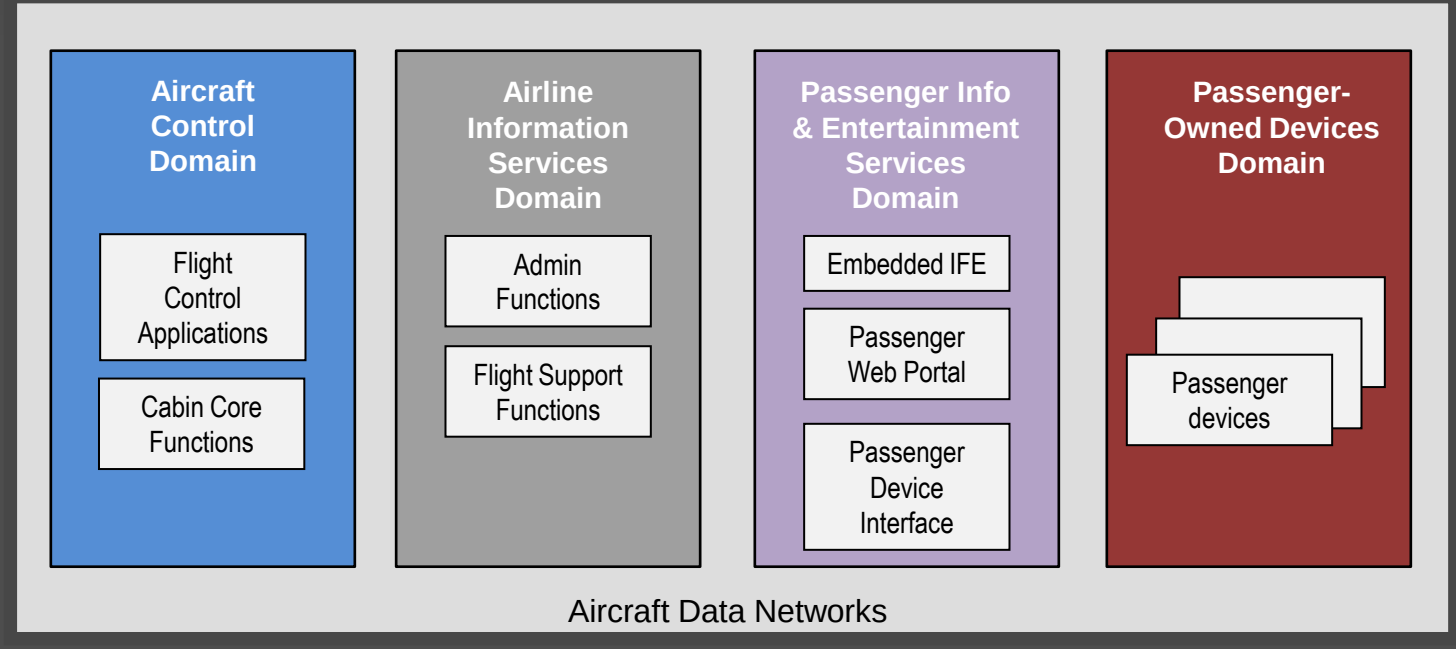
Multi-Level Secure IMA



© 2016 Wind River. All Rights Reserved.

MILS Integrated Modular Avionics use cases

IMA Domain Consolidation



© 2016 Wind River. All Rights Reserved.

Conclusions

- MILS has enabled development of high-assurance multi-level secure systems
  - At affordable cost compared to monolithic approaches
- Multicore is a disruptive technology
  - Impacts software programming models
  - Provides additional security challenges
- Multicore provides potential for MILS systems with increased performance throughput, including:
  - Cross-Domain Systems
  - Multi-Level Secure IMA
  - IMA Domain Consolidation

© 2016 Wind River. All Rights Reserved.

WIND

AN INTEL COMPANY

© 2016 Wind River. All Rights Reserved.