

Using a Separation Kernel to add Military-Grade Security to Legacy Systems

Security is fast becoming a prerequisite in today's software systems and nowhere more so than when dealing with software reuse. A challenge for the software designer is how to integrate modern military-grade software programs into legacy software designed long before security standards were predominant in system requirements. The panacea: virtualization and particularly the secure separation kernel.



Author: Stuart Fisher, LynuxWorks, Inc.

Traditional, non-networked computers are secure from others in the system because of the physical separation that exists between them (Figure 1). Sometimes physical barriers are put in place to prevent unwanted user access in terms of a lock-and-key approach.

Many modern software systems are designed with such tight project time restrictions that redesigning existing software from scratch is almost impossible. To limit engineering costs and to meet project schedules, it is common practice to see a significant amount of software reuse in many of today's software projects. This, however, poses a problem for architects trying to incorporate modern software security requirements into a code base with no concept of such standards.

Most new military systems require some level of security consideration. In some systems, this may be so stringent that formal certification is required. Attempting such a certification on legacy software would be extremely costly and in many circumstances is unachievable. One solution to this problem is to utilize advances in software virtualization techniques, and particularly a separation kernel.

Virtualization: Truth versus misconception

Software virtualization has long been understood as a way of hosting multiple Operating Systems (OSs) on a desktop computer. In recent years, we have seen virtualization migrate into the embedded realm and start to influence markets such as automotive, medical, and industrial, as well as the more traditional aerospace and defense markets.

In most situations, software virtualization is used to address the need for hardware consolidation where multiple systems are combined onto a single hardware platform performing multiple functions. This integration is further complicated by mixing legacy software on the same platform as new design and utilizing the separation between those software components to enforce security in the system.

A common misconception in the software world is that virtualization implies separation and that just because a platform utilizes virtualization, then its software subjects must be separated. In the security world, it is well understood that this is not the case. And many virtualization architectures and products on the market today cannot guarantee

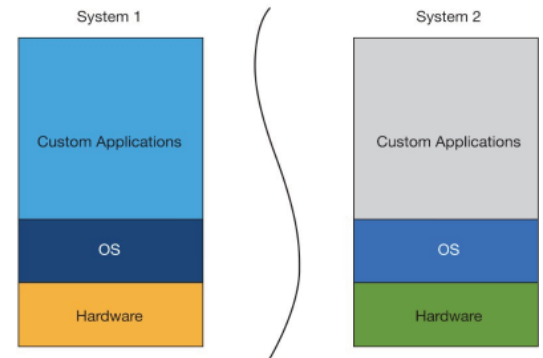


Figure 1: Security enforced by physical separation

any level of software separation and, therefore, are not candidates for military systems requiring any level of security certification.

These architectures would not be suitable as a solution to the problem being addressed here. In this scenario, the separation kernel is quite different from traditional hypervisors. The separation-kernel hypervisor, such as LynxSecure from LynuxWorks™, not only allows multiple guest operating systems to run on the same hardware platform, but it also guarantees that those guests are separated and cannot affect each others' functions.

Not only does a separation kernel separate the guest operating systems, it additionally separates the physical

devices and information flow between the various guests. A software designer has the ability to dictate which operating system has visibility of certain board devices and which guest operating systems are allowed to communicate with each other. It is the implementation of such communication paths that facilitates interpartition communication between guests. With such a path, the guest has no visibility or knowledge of its peer's existence.

Virtualization: A closer look

Using the separation kernel as a base technology, the software designer can now guarantee that one operating system cannot affect another or access certain board devices.

As Figure 2 illustrates, the Windows® subject is running legacy application code in a "contained" Windows environment. The OS has no knowledge that it is running on a separation kernel or that another operating system is running on another core on the very same processor. The second operating system is designed to be the secure gateway and employs complex security software to protect the system from the outside world. Any data coming from the public network is first analyzed by the secure partition, and only if it is deemed secure does it make its way via interpartition communication to the Windows partition.

Using this approach, the software designer has the flexibility to design the secure partition from modern software principles while the legacy Windows OS is completely unchanged. The Windows OS simply sees the interpartition communication path as a connection to

the outside network and has no knowledge that an intermediate software "guard" was analyzing the data and adding a level of software security to the non-secure legacy software.

This premise could indeed be extended to any number of theoretical guest operating systems, each performing a dedicated role in the overall system. Some of these guests might comprise legacy code, while others comprise newly developed code. Systems in the field today already employ such technologies in modified designs. Products such as secure separation kernel hypervisors not only provide a COTS methodology, but they also enable the military market to use modern military-grade software technologies alongside legacy software.

Virtualization melds legacy and secure apps

In conclusion, virtualization and particularly separation kernels are not just tools to allow users to host multiple operating systems on a desktop; they are also valuable technologies enabling system architects to extend the usability of legacy systems alongside but separate from more modern, secure military systems. One virtualization technology, as mentioned, is the LynxSecure separation-kernel hypervisor, certifiable to the highest level of robustness and capable of hosting both paravirtualized and fully virtualized guests including the Windows, OpenSolaris, and Linux® operating systems.

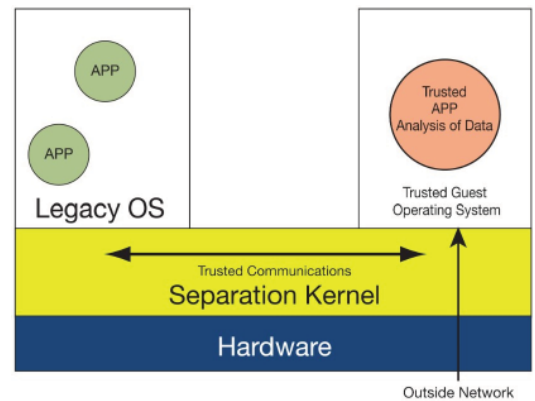


Figure 2: Using a separation kernel



Stuart Fisher is a product manager for LynxSecure at LynuxWorks, Inc. He has more than 15 years of experience in the embedded market, both in engineering roles and customer-interfacing positions. Stuart is based in Sutton Coldfield, England and is a graduate of the University of Coventry, where he earned a Bachelor of Engineering degree in Computing and Electronics.



1.800.255.5969



LynuxWorks, Inc.
855 Embedded Way
San José, CA 95138-1018
408.979.3900
408.979.3920 fax
www.lynuxworks.com

LynuxWorks Europe
50 Broadway
London SW1H 0RG
United Kingdom
+44 208 906 9506
+44 208 906 2338 fax

©2011 LynuxWorks, Inc. LynuxWorks and the LynuxWorks logo are trademarks, and LynxOS is a registered trademark of LynuxWorks, Inc. Linux is a registered trademark of Linus Torvalds. All other trademarks are the trademarks and registered trademarks of their respective owners.

All rights reserved. Printed in the USA.