Using Virtualization to Implement a Scalable Trusted Execution Environment in Secure SoCs

Document Number: MD00993
Revision 01.00
December 3, 2012

Abstract

The digital revolution has been characterized by an explosion of new applications that take advantage of today's high-speed broadband architecture. These include applications such as HD video streaming, gaming, VPN, social media, cloud computing and storage, and machine-to-machine (M2M) communications between embedded devices across both wired and wireless networks. Productivity and consumer entertainment has never been higher with these new applications, but this era has brought with it an increased risk of security compromises including piracy, IP theft and espionage, credit fraud and identity theft, and homeland security breaches, to name a few. Security has therefore come to the forefront of embedded system design.

Static-based approaches for embedded system security which define secure and non-secure zones by partitioning separate hardware subsystems for each zone have been effective so far. However, more scalable and cost-effective approaches are required to address the needs of newer devices running multiple applications over several secure zones. This paper examines the use of virtualization for creating the requisite scalable trusted execution environment for secure embedded systems. It also provides an overview of MIPS' existing and forthcoming solutions for virtualization-based security.

1 The Need for Better Security in Connected Embedded Devices

Throughout the evolution of the internet, we have seen ongoing innovation as new services and applications drive growth in network capacity, and the resulting excess capacity then spurns the development of new services and applications. This trend continues in alternating cycles. It was only a few years ago that the telecom industry was talking about the "bandwidth glut" and the lack of the killer app that could consume the tremendous unused capacity in dark fiber. These discussions are well behind us, and we are clearly once again in the "services driving bandwidth growth" phase, with a plethora of killer apps including video on demand, network gaming, video conferencing, e-commerce, telecommuting, cloud storage, social media and others. Broadband connections to the home have risen dramatically, and the number of connected devices, including smart phones, tablets, gateways and set top boxes (STBs), continues to grow at a blistering pace. Adding to this evolution and bandwidth consumption will be connected cars and the "internet of things" where even municipal and industrial machines will be increasingly connected to ubiquitous broadband IP networks.

With this growth in the number of applications and connected devices comes a need for increased security. Security related concerns are skyrocketing among service providers, consumers and even governments as usage increases. Digital Rights Management (DRM) is a critical concern for film and TV studios in protecting their content from piracy—a problem that already leads to billions of dollars in losses every year. Traditional STBs have been the primary means for delivering video to consumers, but the risks have increased with more people streaming video to their smart phones, tablets, gaming systems and non-traditional over-the-top (OTT) STBs, which connect directly to broadband links.

Telecommuting has also increased with the availability of fat pipes on both wired and wireless broadband connections, which make working from home practically seamless from an IT standpoint. Several enterprises even support the use of mobile enterprise applications deployed on remote laptops or tablets. Obviously, the risk of the leak of confidential enterprise information is heightened as users mix work and personal use on their connected devices. We are also seeing increased deployment of smart gateways, which provide not only a broadband connection for the consumer, but also provide utility information—such as gas and electricity consumption—to a public utility company.

Governments are increasingly concerned about homeland security, as would-be hackers may be able to access public utility IT systems if the customer premise equipment (CPE) is not secure. There are many other examples, but the point is that there has never been more risk for compromised security than today, with the widespread use of connected devices. Therefore, embedded devices must be able to effectively and reliably isolate secure applications from non-secure applications while meeting the appropriate levels of functionality, performance, cost, and power consumption.

2 A Generalized Framework for Embedded Security and Common SoC Implementation

Many of today's SoCs used in connected embedded devices are designed to implement some kind of security framework, like that shown in Figure 1. This section describes each key element of a secure SoC.

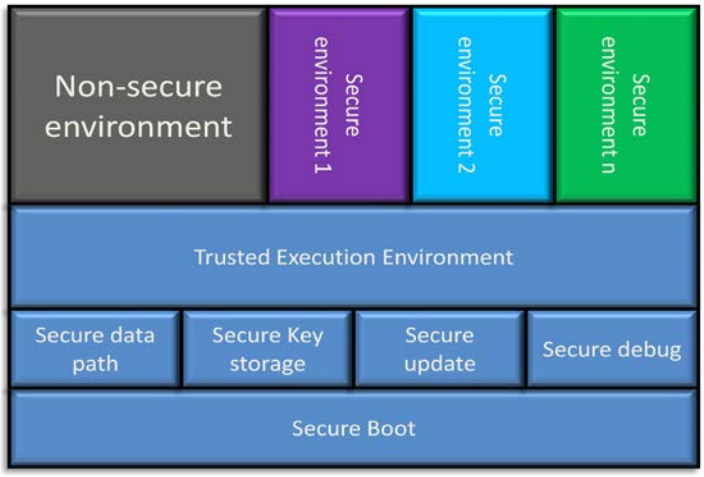

Figure 1 – Generalized framework for a secure SoC

Secure boot: The secure boot is the so-called "root of trust" which is intended to be tamper proof. It is typically implemented as a read-only memory (ROM), which holds the initial boot code loaded upon device reset. Once the root of trust is up, it authenticates the signature of a boot loader for the main software bundle, then loads the signature upon positive authentication. For example, in an STB, all of the software, encrypted with a private key, may be bundled into a flash device. This software is loaded only after it has been validated by the secure boot using the public key stored in the one-time programmable (OTP) memory on the SoC. What follows next is a hierarchical loading and verification process for firmware, trusted execution software, operating system and application software, where the lower layers are loaded and then authenticate the layers above them.

Secure Key Storage: This refers to the OTP area where secure assets such as public keys and any other keys for DRM are typically stored. For example, a video-on-demand application such as Netflix might store its public key in the OTP to decrypt the content. Secure boot and secure key storage are the first things you will need in order to create a secure SoC.

Trusted Execution Environment: After the boot loader has been successfully loaded and authenticated, a software layer called the Trusted Execution Environment (TEE) is loaded. The TEE manages and controls access to a set of lower-level software modules that together allow for a secure environment. These sub-modules include secure keys, secure data paths, secure update, and secure debug. The TEE, which allocates resources and prevents non-secure applications from accessing secure blocks, is essentially the gatekeeper to the underlying hardware resources. For example, the TEE in an STB would be responsible for ensuring that non-authorized applications don't access key assets such as video codecs or locations in memory that may hold unencrypted secure information.

Secure Data Path: Secure data path is the sub-entity that ensures high-value assets such as codecs are only accessed by authorized entities.

Secure Update: This is the entity that allows for secure updating of system software by authenticating and managing any software update request from the upper layers.
Secure Debug: The secure debug module ensures that the JTAG ports are secured against unauthorized access. Usage of the JTAG ports may implement an authentication system such as passcodes.

From an implementation standpoint, many SoCs implement a static, dual-processor system similar to that shown in Figure 2.

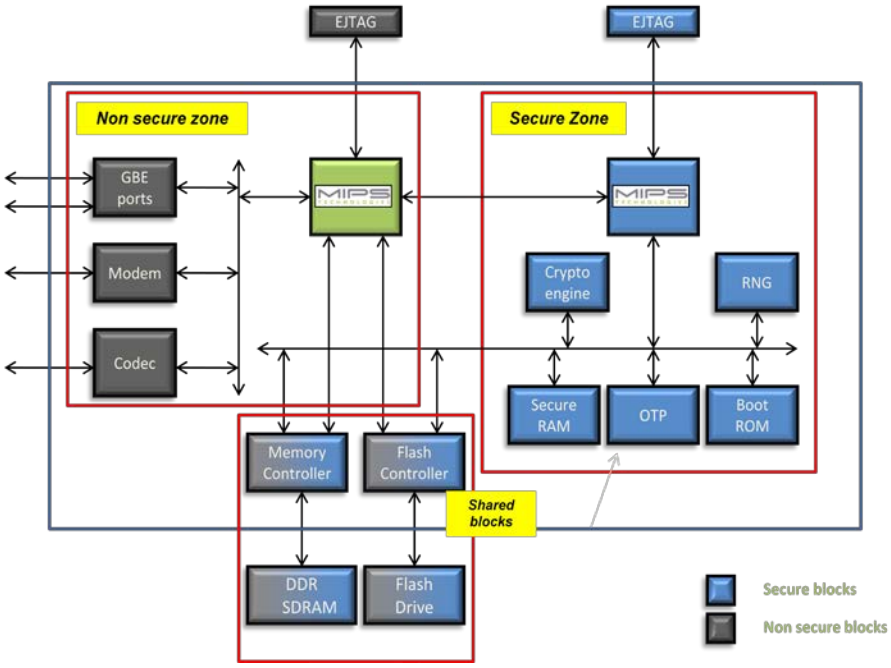

Figure 2: SoC subsystem implementing static security

In such an implementation, one processor subsystem resides in the non-secure zone and another processor is used for the secure zone workloads. Located in the secure zone are all the secure assets such as the codecs, crypto engine, secure memory, OTP, boot ROM, and secured debug port. The non-secure zone would be running the non-secure system OS and application software. Static security implementations are common today and provide a high level of security. Many safety-critical applications, such as avionics, implement static security SoC subsystems because of this.

3 The Scalability Challenge and Virtualization as the Solution

The approach of leveraging more CPUs to implement additional secure zones is a workable solution. However, it is not the most cost- and power-effective way to address the matter, since it means that more secure zones are needed in a single device. In mobile devices, the ability to address security while keeping down power consumption and cost is of the highest priority. Therefore, it is critical to use a solution which scales the number of secure zones in an embedded system in a manner that is not excessive in terms of power and area. This is where virtualization fits in.

Virtualization is a technique for creating multiple, secure execution environments for guest operating systems and applications over a common shared hardware resource such as a CPU subsystem. Figure 3 is a diagram that shows a generic virtualized system. The core element is the

hypervisor, which is a small body of code that sits above the hardware, and serves as the trusted execution environment. It manages the privileged resources by defining access policies for each execution environment. This effectively allows for the creation of multiple logical execution environments called Virtual Machines.

A hierarchy of memory management units (MMUs) is used for the isolation of applications, operating systems and the hypervisor. Specifically, a Guest MMU is managed by the Guest OS to isolate guest applications/users while a Root MMU is managed by the hypervisor to isolate Guest OSes. Only the hypervisor is allowed to interact with secure code.

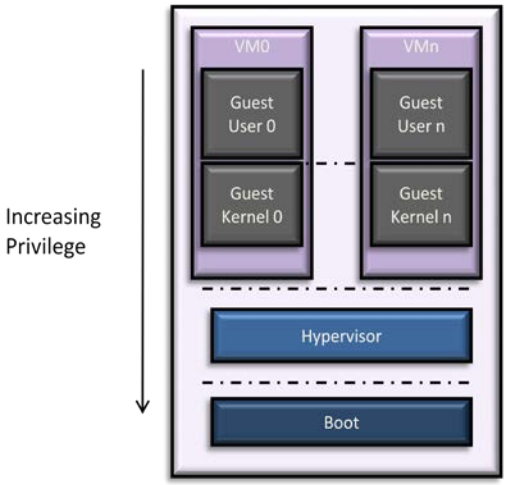

Figure 3 – Virtualization components

In addition to security, virtualization also allows for increased system reliability. Because of the isolation between virtual machines or zones, the other operating systems and applications in one virtual machine can continue to run even if another virtual machine crashes in the same hypervisor entity. Another noteworthy point about virtualized systems is that there is no need to address all security holes of a big OS and/or multiple OSes as long as the hypervisor is secure. This reduces the number of potential breaking points, and hence reduces concerns for the system designer.

There are different ways to implement a virtualized system. One approach is called Trap and Emulate, which implements unmodified OSes on top of a hypervisor. This requires minimal design effort to implement on existing CPUs, but delivers very low performance. A higher-performance approach is called para-virtualization. Para-virtualization improves performance by optimizing the interaction of the OS and the hypervisor. There is reasonable effort required to customize the OSes, but this allows for better performance and can also be used in existing CPUs. Para-virtualization is a reasonable approach for retrofitting a scalable security solution into deployed embedded systems that are not due for additional hardware updates but require a trusted execution environment. A third approach is hardware-assisted virtualization, which delivers increased virtualized system performance through changes in the CPU architecture, without requiring any changes to the guest operating systems.

Using virtualization, a dynamic secure system could be created as shown in Figure 4. The biggest difference versus a static system is the elimination of the second CPU. The non-secure zone and the secure zone are now implemented in two virtual machines running on the same CPU (on the left side of the diagram). The assets within the secure zone block are still secured, but access to these assets is now managed by the hypervisor, which acts as the trusted execution environment. Only the application(s) which are granted access to the secure resources by the hypervisor can actually get to them. By using only one CPU instead of two, designers can save area and power. Multiple secure zones can be implemented in the CPU as needed via virtual machines. Additional cores can be added as needed to accommodate other zones, or to boost performance. However, a single CPU may provide sufficient horsepower depending on the application.

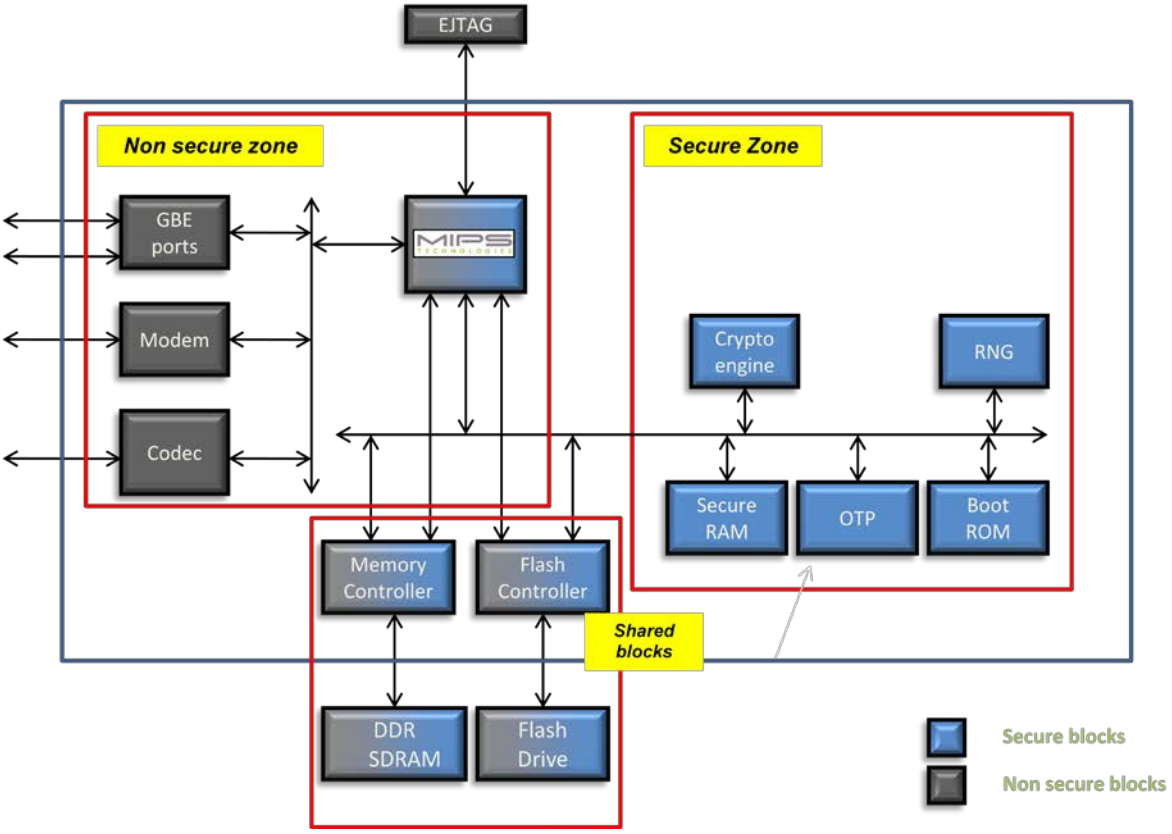

Figure 4 – Dynamic security system using virtualization

As an example, an SoC for an Android tablet might look like the functional subsystem shown in Figure 5a. Since there is only one CPU, a hypervisor is needed to ensure that unauthorized applications do not access the secure assets in the secure zone. Figure 5b shows how the applications and guest operating systems could be mapped to virtual machines running on the hypervisor. A video-on-demand application like Netflix running on an Android OS could be mapped to the first, non-secure virtual machine. To protect secure assets—such as the crypto engine and keys—from being accessed by unauthorized guests, a second virtual machine could be instantiated to run the DRM scheme plus a security client, which serves as a mailbox to the non-secure side.

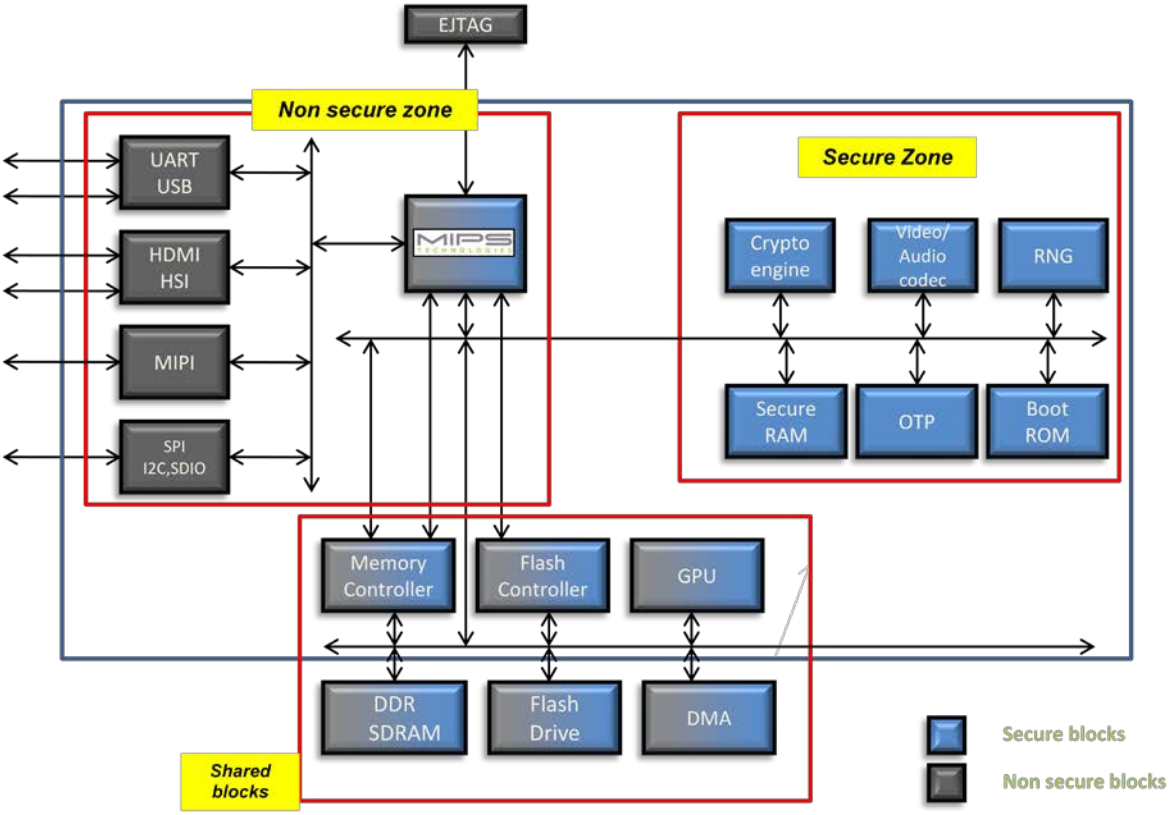

Figure 5a – Tablet SoC with hypervisor-based TEE

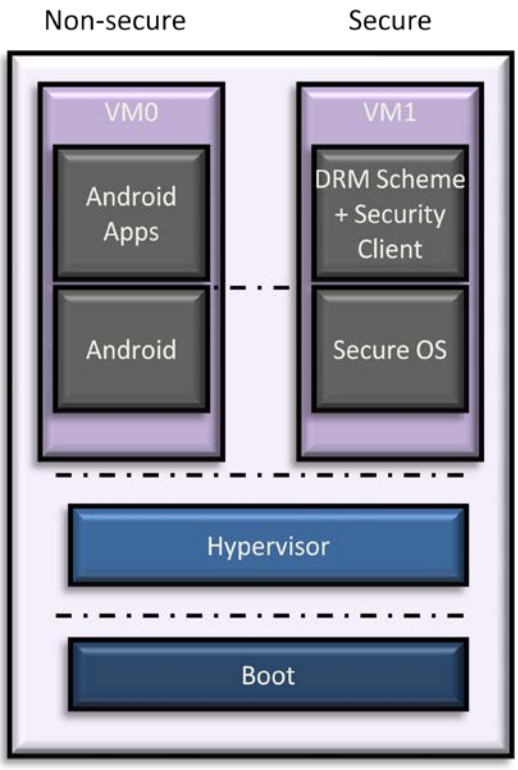

Figure 5b –Virtual machine application mapping for a tablet SoC

To further highlight virtualization's breadth of use in securing embedded applications, we can also look at the emerging smart gateway, which serves as the hub for connected smart home applications. As the gateway increases its intelligence and connectivity to critical network resources such as those within utility companies, it becomes even more important to have a trusted execution environment in the residential gateway SoC. An example residential gateway CPU subsystem is shown in Figure 6a. Figure 6b demonstrates how a first virtual machine could run the smart home application software plus the residential gateway software on the non-secure side. A second virtual machine could be used to run the smart home security application and the security client.

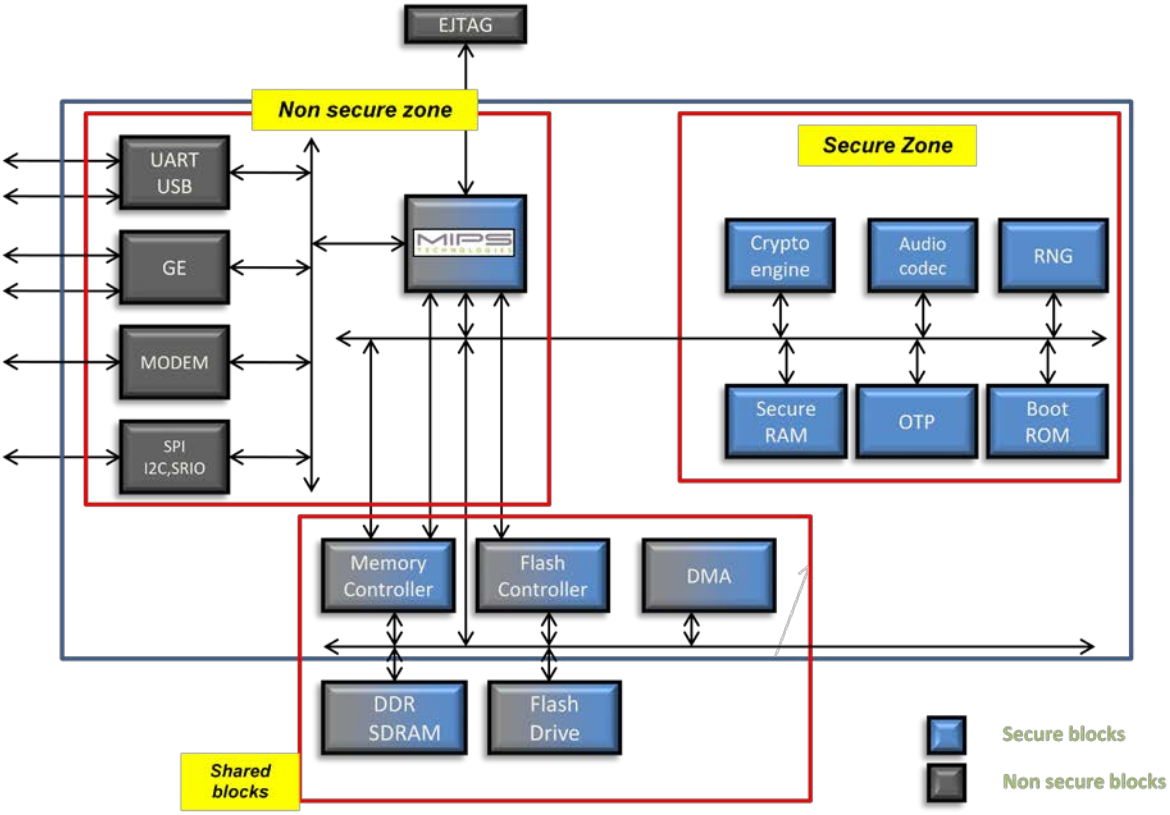

Figure 6a – Smart gateway SoC with hypervisor-based TEE

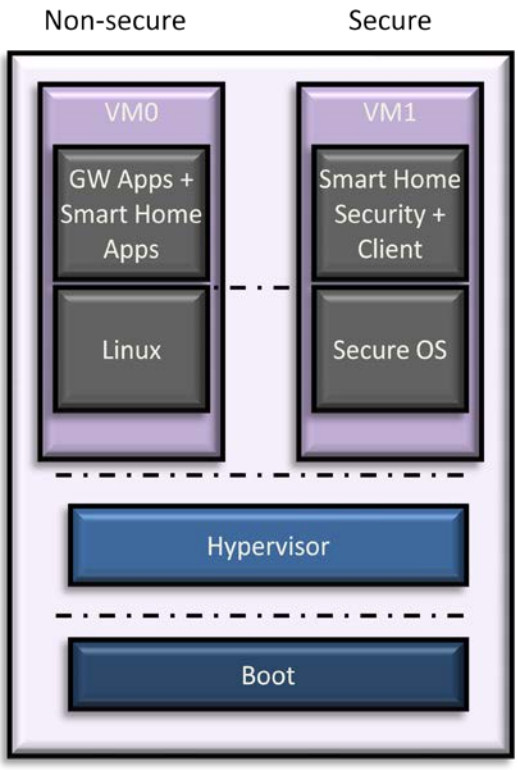

Figure 6b – Virtual machine application mapping for smart gateway SoC

4 MIPS Virtualization Support

Para-virtualization Solutions

Customers looking to implement para-virtualization on MIPS devices can utilize hypervisors from third party partners such as SYSGO AG, now part of Thales Group. Hypervisors such as SYSGO's PikeOS perform a baseline set of functions as well as value-added functions including hardware abstraction, inter-partition communication, module configuration, resource and time partitioning, and health monitoring. MIPS partners also make available popular operating systems that have been optimized to work with hypervisors. Para-virtualization is an excellent option for existing SoCs that may need to be retrofitted with cost-effective security mechanism that could be deployed quickly.

MIPS Release 5 Architecture with Hardware-Assisted Virtualization

With the demand for secure solutions, and corresponding use of virtualization solutions expected to increase, MIPS has implemented hardware-assisted virtualization support in its Release 5 architecture. The MIPS Virtualization (VZ) module is a highly-scalable option that provides a number of capabilities, including enhanced security features and support for multiple operating systems. The MIPS VZ module is a simple and flexible hardware-based solution that satisfies varied security requirements with limited or no performance impact. A complementary whitepaper which describes the Virtualization features of the Release 5 architecture in more detail is also available on the MIPS website for download.

5 Summary

Today's connected embedded devices are highly-integrated and are capable of simultaneously supporting a multitude of applications. These applications have enhanced the consumer's entertainment experience and work productivity with an anytime and anywhere usage scheme, but with that has come the exponentially increased threat of security compromises. Requirements for SoC security have also escalated to meet the challenge, but scaling the security capabilities while managing cost and power consumption is key.

Virtualization is beginning to emerge as a viable, cost- and power-efficient solution for creating trusted execution environments in secure SoCs for connected applications. Virtualization has been widely deployed in PC/server systems, with a focus on maximizing server utilization and reducing OpEx. However, only recently have we seen it deployed in security for consumer-class embedded systems.

MIPS believes the usage of virtualization will increase substantially in the future with even more connected device types and applications. In support of this vision, MIPS Technologies delivers a complete range of virtualization solutions. This includes hardware-assisted virtualization with the Release 5 architecture and para-virtualization through third party partners. Learn more by downloading the Release 5 specification and MIPS Virtualization white paper. Contact MIPS to discuss options for virtualization-based security in your secure SoC design.