Trusted Computing on Heterogeneous Embedded Systems-on-Chip with Virtualization and Memory Protection

Marcello Coppola
ST Microelectronics
Grenoble, France
marcello.coppola@st.com

Miltos D. Grammatikakis and George Kornaros
Technological Educational Institute of Crete
Heraklion, Greece
{kornaros, mdgramma}@cs.teicrete.gr

Alexander Spyridakis
Virtual Open Systems
Grenoble, France
a.spyridakis@virtualopensystems.com

***Abstract*—The paper examines the architecture of a secure and trustworthy cloud platform, which ensures strong logical and physical security on the client devices using a two-layer security mechanism: a) a hardware security module located on the SoC of the client device that protects incoming and outgoing communications (e.g., to/from an external memory) against physical attacks, and b) system software and hypervisor extensions that isolate virtual machines from one another and from the underlying hardware in order to protect against logical attacks.**

Keywords-cloud computing; confidentiality; integrity; multicore SoC; protection; security; virtualization.

I. INTRODUCTION

A new era is emerging in consumer, industry, and government areas, where traditional consumer and mobile devices are replaced by intelligent, next generation systems, such as smart phones, smart TVs and smart tablets that provide innovative services, such as social networking and on-demand multimedia (e.g., Netflix* [8]), by connecting to the cloud. Meanwhile, content providers increase the availability of large-scale, high-quality libraries of web data with text, images, sounds, videos and animations. The technology races towards new generation, powerful, complex, smart devices promotes convergence of traditional video and Internet-based content deployed in cloud computing infrastructures and increases the possibility of security breaches.

For example, devices, such as Intellectual Property (IP) set-top boxes, residential gateways or media players, now provide a multitude of services, such as graphical user interfaces, digital rights management, secure transcoding protection, network provisioning and payment. Each service finds its physical representation in a mixture of hardware and software components, ranging from small security-critical software stacks running on basic processors or accelerators, up to commodity operating system (OS) on complex application processors. Since each of these highly heterogeneous software stacks uses sensitive data that must be protected, individual services must collaborate to enable global system security [5] [11]. This leads to a significant increase in complexity and associated development costs.

Security solutions for end-users (individuals, companies) connecting to the cloud using client equipment are of utmost concern in the era of cloud services and applications [1]. Cyber-secure architectural solutions for cloud environments must offer ways to fully secure system and end user applications and services against cyber-criminal end-users, even for the components that will run on the client side. Today the lack of appropriate isolation of source code and data among trusted and untrusted applications is one the main challenges in building a secure architectural solution. On the other hand, offering trustworthy cloud computing services that would prevent from rogue administrators spying or altering end user data and computations requires significant hardware and software modifications in data center architecture. This implies that on the end user side, there is no trust to the cloud provider, especially if the end user stores confidential info. Therefore, a viable and economical solution is to enhance the security level of the connected smart device when accessing the cloud. This new idea could speed up utilization of cloud infrastructure by connected devices and allow service providers to trust sensitive computations performed by end users and consequently delegate processing tasks to them.

This paper describes work in progress that aims to provide a viable solution towards protecting the integrity and confidentiality of sensitive data (e.g., movie, photo, e-book) and software applications in a modern cloud infrastructure where approved devices are connected to the cloud. This work targets protection from two kinds of adversaries: (i) rogue applications such as virus, Trojans possibly launched by the user himself, (ii) physical adversaries such as probing, spying at or tampering with the communication link connecting the device to the external cloud environment.

Section II considers the current state-of-the-art in System-on-Chip (SoC) virtualization including existing memory protection strategies. This section lays out the path towards presenting the TRESCCA security approach in Section III. Finally, Section IV concludes the paper.

II. VIRTUALIZATION AND SECURITY

On top of a hardware platform, we have the software stack, including the OS, the middleware and the application layer. Security of the device that runs applications from different sources is usually under the responsibility of the OS. The OS uses software (e.g., virtual memory, file permission, memory protection) and ad hoc hardware mechanisms to isolate different applications sharing common physical and logical resources, such as software libraries, services and resources, e.g., printers, graphics accelerators. The complexity of modern OSs (large number of code lines, developed by different groups) creates different security vulnerabilities resulting from software misbehaviors. These are exploited by a cyber-criminal, who attempts to subvert the security mechanisms supported by the OS and get control of the device and data. For instance, overwriting data or function pointers, dynamic memory allocation (double-freeing, referencing or writing to free memory, zero-length allocations, and buffer overflows are well known techniques used to bypass any security protection imposed by the OS.

Vendors are now using virtualization technology to isolate physical resources from applications and platforms that use them. This is performed by introducing the virtual machine (VM) concept that serves as a guest software environment that supports a stack consisting of an operating system (OS) and application software. Each VM is independent of other VMs and uses the same interface to processors, memory, accelerators, and I/O provided by a physical platform. VM isolation provides the means to regulate application access to computational resources, thus enabling malware detection capabilities. Isolation is achieved by inserting a hypervisor layer between the operating system and the hardware. This enables the hypervisor layer to govern all interactions that take place between the OS (and the layers above it) and hardware [4] [7].

In full virtualization, the hypervisor provides the same hardware interfaces as those in the physical platform, hence the guest OSs and applications do not need to be modified. Since full virtualization increases information sharing among different system layers, security maintenance becomes very complex. Thus, NIST has proposed security management recommendations that involve the host OS (if applicable), the hypervisor and the guest OS [9]. NIST best practices (policies and checks) for a secure hypervisor layer involve installing updates, monitoring, restricting access via authentication, encryption and integrity mechanisms, disconnecting/disabling unused hw/sw components and performing clock synchronization [10]. The specified practices affect hypervisor configuration, initiation, design and planning, implementation, operation, maintenance and disposition and ensure that data access and transmission threats are thwarted.

A. Embedded Virtualization and Security

Mobile platforms and set-top boxes are in the middle of a global transition period in which client devices manage to support high-level operating systems and middleware, quickly moving from a close or walled garden limited environment to a setting where a walled garden has to coexist with an open one. In this new scenario, devices are able to run any third-party application that may or may not be certified by the operator. In this context, it is crucial to ensure that third party applications cannot break security. Otherwise, if isolation is broken, sensitive content could be easily stolen or edge devices could be used as a Trojan horse to break cloud security. Hypervisors would allow vendors to isolate important trusted services (e.g., billing, authentication, phone service) from the open operating system layer and run them in isolated, tamper-proof virtual machines (VMs). Thus, trusted services are not affected even if the open environment is compromised.

Traditional virtualization technology resolves isolation of different applications at the processor level, but suffers from non negligible drawbacks [7]. Indeed, it allows sharing of processing and shared memory resources efficient and secure on homogeneous SMP architectures that can be controlled on a common trusted basis. However, it is not secure for heterogeneous shared-memory multiprocessor systems-on-chip (MPSoCs). In fact, most connected smart devices architectures are heterogeneous, including different islands of computation such as GPU, DSP and hardware accelerators. Islands of computations cannot natively support virtualization, since they lack memory management units, and often do not offer inherent ways of establishing privilege levels. Therefore, applications running in such systems are able to access the whole address space, breaking the required isolation assumption imposed by virtualization. In order to address these issues, security hardware extensions to processor and interconnects are being considered.

A few years ago, bi-partitioning techniques introduced in ARM's TrustZone [3] extended the ARMv6 architecture by adding the concepts of "secure" and "non-secure" states and a "secure monitor mode" used for switching between the two. In addition, the AMBA3 AXI has been extended with two new signals (ARPROT/AWPROT) that indicate whether the respective read/write transaction is secure or non-secure. Nowadays, binary bi-partitioning cannot meet the security requirements of cloud-connected devices. Moreover, TrustZone technology cannot protect against bus probing which can be used to attack the software stacks.

MPSoC security must be addressed by a platform-wide protection mechanism covering the full communication infrastructure, instead of a processor-centric mechanism [12]; similar approaches have also been proposed in [5] [11]. The proposed concept defines a protection domain as a set of specific access rights to a shared address space and maps each software stack to a specific domain. Notice that software stacks may have right overlaps between them.

In order to make the security check the proposed approach may suffer from long latency, especially if there is a miss in the local permission look-aside buffer, and the missed entry has to be loaded from external memory. Thus, due to the granularity of the security checks, silicon cost is unacceptable for embedded devices.

The basic concept in our approach is to implement a low-cost solution at the Network-on-Chip (NoC network interface. With ideal distributed co-hosting of several protection domains, software stacks transparently and efficiently share resources (processors, memory and peripherals) issuing memory accesses through Direct Memory Access (DMA) controllers.

III. The TRESCCA Approach

The TRESCCA architecture secures critical data in a fully end-user transparent way, without storing information in centralized pools that define an attractive attack point [13]. TRESSCA consists of a CPU cluster, on-demand media accelerators and storage interconnected in a heterogeneous shared memory MPSoC via a complex NoC (STM's Spidergon STNoC). Each CPU cluster is a symmetric multiprocessor (SMP) hosting OS execution.

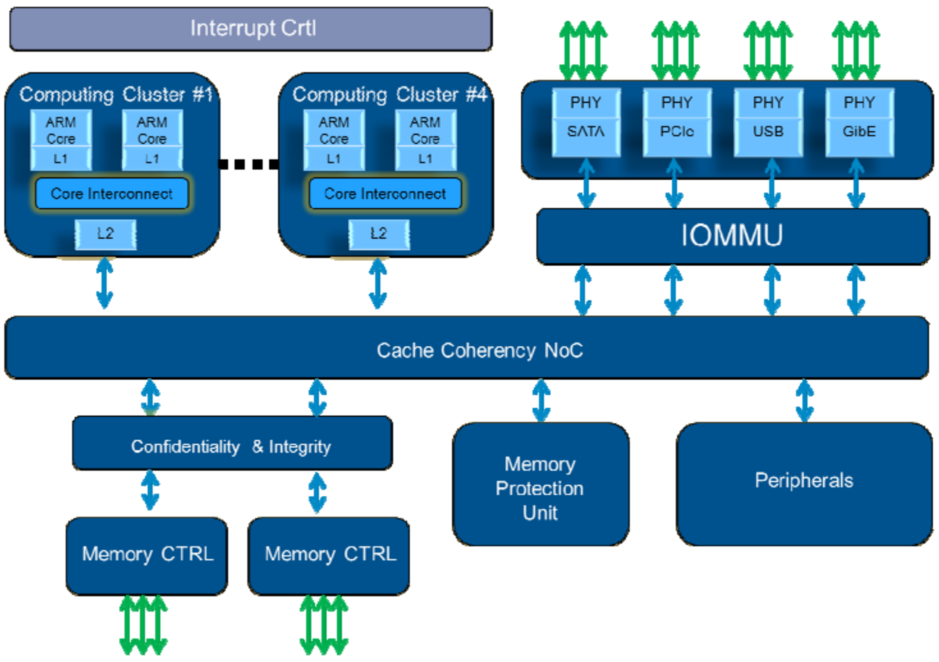

Figure 1. TRESCCA architecture with Hardware Security Module (HSM)

A. *Secure Information Processing*

TRESCCA introduces a novel security infrastructure that aims to protect the confidentiality and integrity of sensitive software against two types of adversaries:

- Logical adversaries like rogue applications: viruses, Trojans or malware launched by the end-user.
- Physical adversaries like the end user himself, with complete physical access to the system. For instance, the end user can issue a board-level attack by probing the bus between the SoC and its external memory or tampering with a system communication link.

Notice that these two kinds of attacks can also be combined, as has already been done recently against famous game consoles and other consumer equipment.

Protecting the system against logical adversaries will rely on virtualization techniques, while board-level physical attacks will be prevented by input and output data encryption and integrity checking. Both memory protection and virtualization techniques, implemented using hardware and tightly-coupled system drivers, will jointly reinforce a secure hypervisor kernel that isolates critical applications and prevents memory tampering. The following subsections describe how TRESCCA enhances the NoC backbone by extending its network interface and how these extensions help the hypervisor build the required security infrastructure.

B. *NoC Firewall*

The NoC communication infrastructure enforces strong isolation of VM by tagging the underlying transactions. What this means is that a potentially compromised Guest OS in a Virtual Machine cannot access data that is tagged by another VM. Next, we use the term *domain* to refer to an isolated environment in the platform, to which a subset of the shared physical memory is allocated.

Using the virtualization concept, we can create a level of indirection between physical and virtual components. Each physical component is associated to many different virtual instances that are allocated to a domain and are referred to as the domain's assigned components. For modern CPUs, this is possible using hardware virtualization extensions [2], for other components, such as DMA or hardware accelerators, an IOMMU is used [6].

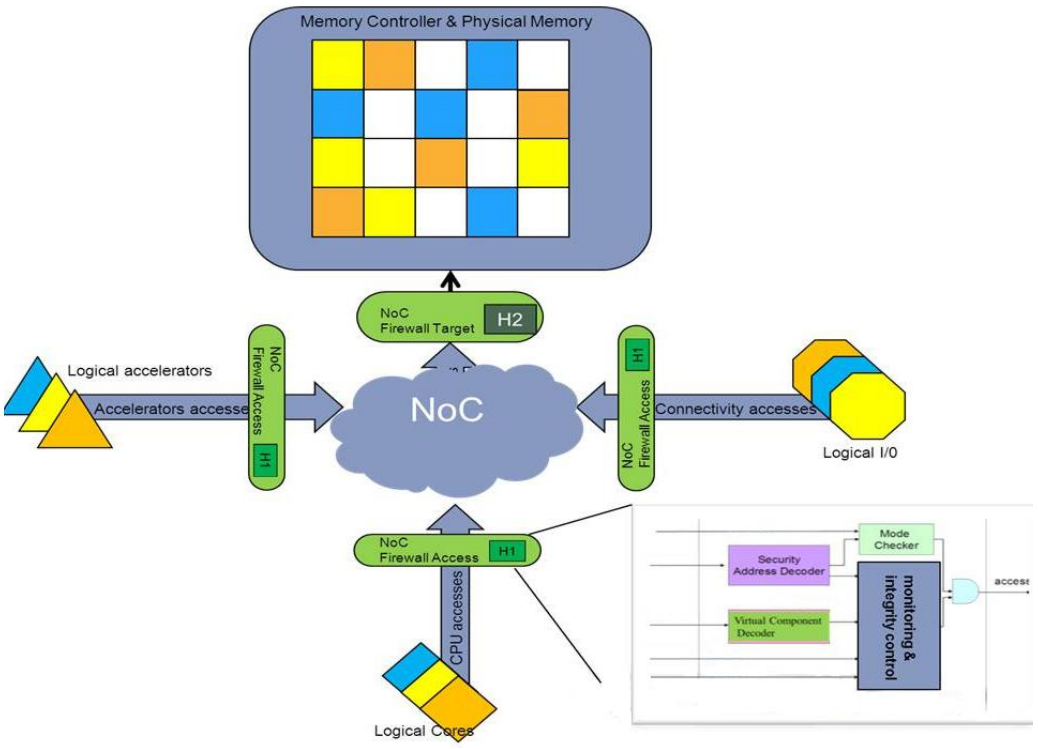

Figure 2. Partitioning of physical memory to different logical domains

Multiple domains can co-exist in a platform and a virtual component (e.g., a virtual CPU) can be mapped to a single domain. True domain isolation is achieved by blocking accesses (read/write NoC transactions) from resources outside the domain to their assigned physical memory; ideally, network routing paths are also balanced across VMs by assigning separate virtual circuits. This implies establishing a set of access rights on different address regions and ensuring that these rules are observed at each network interface. Our solution will be processor-independent, although interrelation to predefined processor privileged levels is desirable (e.g., ARM v7 PL0, PL1, etc).

Each initiator transaction is tagged with a corresponding VM and/or process identifier. The main innovation point for defining the set of access rights for each tuple (VM id, process id, and physical address) is introducing two levels of memory hierarchy. These consist of hierarchy 1 cache at all initiator interface and a hierarchy 2 cache at the target interface (resp. H1, H2 in Figure 2). In case of H2 miss, the NoC Firewall target interface is responsible for fetching the required entry from the physical memory containing the permission tables shared by the different NoC Firewall access points. A scalable NoC Firewall will enable flexible and efficient assignment of virtual components to an arbitrary number of domains, proving low latency and power-efficiency compared to past research, such as [12]. Moreover, by policing the NoC Firewall access point at the initiators, we would be able to detect and subvert Denial-of-Service attacks, where malicious code attempts to saturate the NoC through massive unauthorized accesses.

At the physical level, NoC Firewall and associated cryptography will ensure that all transactions between the SoC and its external environment are protected through domain isolation, confidentiality and integrity [5]. Thus, it will be infeasible for an adversary to spy or alter sensitive data crossing the SoC boundary without issuing an interrupt.

C. *STNoC Synthesis*

We have synthesized STNoC using STM 32nm technology in order to estimate the area overhead of the NoC Firewall. Assuming 20 domains and a NoC with 80 initiator and 68 targets, a secure AXI read-only interface occupies 23 to 30K gate equivalent (GE), compared to 20 to 28K GE for the non secure case. Similarly an AXI write-only interface occupies 21 to 51K GE, compared to 19 to 49K GE for the non-secure case. Hence, the area overhead is 5 to 14% for read- and 3 to 11% for write-only interface, depending on the precise AXI configuration .

D. *Extended Hypervisor Security*

At the software layer, the TRESCCA hypervisor (KVM) must provide strict isolation by running different VMs on the connected devices. Thus, in our methodology, a *trusted VM* associated to a trusted domain, where data and code encryption is enforced, is assigned as the security master of the SoC resources, excluding any IO components. This VM is responsible for creating and managing domains, for allocating physical memory to all domains and for setting up the necessary virtual and physical address mechanisms. For example, this VM can master the rendering functionality of a display, to avoid any malware execution that captures authentication data. Security is further enhanced by a set of approved applications that software encrypt (possibly via an on-chip hardware accelerator) communication with the external memory, provide integrity checking and dispose any unused network connection. This way each application is completely isolated and external attacks are not possible.

In addition, the hypervisor defines a *secure VM* managing all closed or corporate “walled garden” applications (cf., set-top box example). The secure VM is associated to a secure domain that may include I/O accelerators and provides services to connect to the external world, e.g., to an untrusted VM. The main difference from an application running on the trusted VM is that these applications can communicate through a firewall to the cloud for additional computing power and/or storage.

The remaining VMs can execute untrusted applications and connect to the external cloud environment. In these VMs there is always a risk that a downloaded application exploits security vulnerabilities. Therefore, mandatory monitoring and integrity control (MIC) protocols at the underlying NoC Firewall (see Figure 2) ensure that security policies are uniformly enforced at the hypervisor layer [1]. Our custom MIC hardware extensions are related to software security, similar to mandatory access control (MAC) extensions in SELinux, e.g., the Loki tagged memory architecture [14]. Restricting different workloads through our MIC ensures that viruses and other malicious code cannot spread from one VM or guest OS to another, and data cannot easily leak from an untrusted VM or guest OS to another one even if VMs start to misbehave.

IV. CONCLUSIONS AND FUTURE WORK

Cloud computing is an emerging technology that quickly goes mainstream, making our society increasingly online, with consumers using browsers embedded in mobile devices or modern TV sets to access e-mail and social media. Besides smart phones and TVs and tablets grabbing the headlines, in the near future game consoles, cameras, photo frames, radios, printers and set-top boxes will also be connected to the cloud. Depending on the nature of the threat, cloud security must encompass three components: confidentiality, integrity, and availability. Confidentiality is violated whenever sensitive information is disclosed to any unauthorized entity (human, program, or system). Integrity is violated whenever unauthorized code is executed or unauthorized data is used. Availability is violated when an attacker succeeds in denying services to legitimate users.

The ongoing TRESCCA project develops a lightweight, non-intrusive secure hardware and system software-based infrastructure, that supports multiple domains on top of virtualization technology, in order to realize separation among client’s broadband services (e.g., in Android) and global broadcast services (e.g., in NDS, HbbTV). This client-centric, “walled garden” allows client control over its application code and media content. Moreover, virtualization technology will allow set-top box or smart TV to efficiently execute (and migrate, if necessary) multiple virtual machines enabling hardware consolidation, increased utilization and energy savings. Thus, different middleware and OSs can run simultaneously on a single device, laying the foundations for reducing cost, while promoting interoperability of secure and trustable interactive services and cross-platform application scenarios in heterogeneous virtualized multicore systems. Most project outcomes will be publicly released as open source software. Functional specifications of the architecture currently developed aim to characterize performance and silicon overhead with typical execution scenarios that run on top of an extended open source, secure KVM hypervisor.

ACKNOWLEDGMENT

This work is partially supported by the EC through FP7 collaborative project TRESSCA (GA No. 318036).

REFERENCES

[1] A.M. Azab, "New system security mechanisms for cloud computing," PhD Thesis, NCSU, Dept. CS, 2011.

[2] ARM, "Virtualization is coming to a platform near you," see http://www.arm.com/files/pdf/System-MMU-Whitepaper-v8.0.pdf [retrieved:4/2013]

[3] ARM, "TrustZone: security foundation," available from http://www.arm.com [retrieved: 4/2013]

[4] G.W. Chow and A. Jones, "A framework for anomaly detection in okl4- linux based smartphones," Proc. Australian Information Security Management Conf., 2(2), pp. 5-10, December 2008.

[5] L. Fiorin, G. Palermo and C. Silvano, “A security monitoring service for NoCs,” Proc. Conf. Hardware/Software Codesign and System Synthesis, New York, NY, USA: ACM, pp. 197–202, October 2008.

[6] G. Kornaros, M.D. Grammatikakis, and M. Coppola, "Towards full virtualization of heterogeneous NoC-based multicore embedded architectures," Proc. Conf. on Embedded and Ubiquitious Computing, pp. 345-352, December 2012.

[7] S. Nanda and T. Chiueh, "A survey of virtualization technologies," Technical Report, SUNY - Stony Brook, 2005.

[8] Netflix, http://en.wikipedia.org/wiki/netflix[retrieved: 4/2013]

[9] NIST, available from http://www.nist.org [retrieved: 4/2013]

[10] K. Scarfone, M. Souppaya, and P. Hoffman, “Guide to security for full virtualization,” NIST standard SP 800-125, January 2011.

[11] G. Palermo, L. Fiorin, S. Lukovic, V. Catalano, and C. Silvano, “Secure memory accesses on networks-on-chip,” IEEE Trans. Comput., vol. 57, no. 9, pp. 1216-1229, September 2008.

[12] J. Porquet, A. Greiner, and C. Schwarz, "NoC-MPU: a secure architecture for flexible co-hosting on shared memory MPSoCs," Proc. Design Automation and Test in Europe, pp. 591-594, March 2011.

[13] TRESCCA project, http://www.trescca.eu [retrieved: 4/2013]

[14] N. Zeldovich, H. Kannan, M. Dalton, and C. Kozyrakis, “Hardware enforcement of application security policies using tagged memory,” Proc. Symp. Operating Systems Design and Implementation, pp. 225-240, December 2008.