2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data Science and Systems

Efficient Asynchronous Communication Between Virtual Machines in Embedded Systems

Rui Wang [12]
1 School of Computer Science and Engineering, Beihang University, Beijing, China
2 Science and Technology on Communication Networks Laboratory, Shijiazhuang China

Libin Xu
School of Computer Science and Engineering, Beihang University, Beijing, China

Yuebin Bai*
School of Computer Science and Engineering, Beihang University, Beijing, China

Zhongzhao Wang
School of Computer Science and Engineering, Beihang University, Beijing, China

Hailong Yang
School of Computer Science and Engineering, Beihang University, Beijing, China

Lijun Zhang [12]
1 School of Computer Science and Engineering, Beihang University, Beijing, China
2 Science and Technology on Communication Networks Laboratory, Shijiazhuang China

***Abstract*—Embedded virtualization systems become increasingly important in the light of mobile computing era. Current OKL4 technology has overcome some limitations of plain virtualization by introducing microkernel approach but also imposes several concerns including incomplete IPC (Inter-Process Communication). With this in mind, in this paper a novel asynchronous communication mechanism to optimize original OKL4 IPC mechanism is proposed. The mechanism is able to effectively deal with concurrent communication requests between any virtual machines (VMs) by generating multiple event channels and providing a complete management mechanism. Further, a shared-memory-based bulk data transfer mechanism based on asynchronous communication mechanism is proposed to enrich OKL4 primitive data exchange approaches. In the end, the promising experiment results prove the feasibility and effectiveness of the mechanisms.**

Keywords—embedded virtualization; OKL4; asynchronous communication; multiple channel; bulk data transfer

I. INTRODUCTION

The core ideology behind virtualization is to implement the mapping from multi-platforms to single physical platform without considering the homogeneous and heterogeneous attributes of them [1, 2], which contributes to making full use of underlying hardware resources and further reducing resources and maintenance costs. Modern embedded systems have been widely used in all aspects of our everyday life. In contrast to traditional embedded systems which used to be simple and simple-purpose, modern embedded systems are increasingly taking on characteristics of general-purpose systems requiring commodity and open APIs [3], but there are still some systems demanding the conflicting requirements such as real-time performance, security and so on. Besides, the advent of multicore chips further motivates the utilization of virtualization technology in embedded systems [4]. Therefore, embedded virtualization technology has been an inevitable trend. PikeOS, Nova, OKL4 and many other virtualization solutions aimed specially at embedded platforms have greatly promoted the development of embedded virtualization technology. Virtualization can provide some attractive benefits to embedded systems, of which the most important one is its ability to provide functional and performance isolation for two types of execution entity: native applications and guest operating systems, which equips the embedded systems with security and reliability [5].

However, there are significant limitations on running VMs (virtual machines), particularly relevant to embedded systems. The subsystems of an embedded system are not independent but are required to co-operate closely in order to achieve the overall function of the system, which demands highly-efficient communications between the subsystems [6]. As the central mechanism provided by microvisor [7], IPC in OKL4 is abstracted as virtual device registers and virtual interrupts for synchronous communication. Although significantly simplified, it still lacks support of multiple channels co-existence and only enables a single channel for a pair of VMs communication at one time.

II. Multiple Event Channel Asynchronous Communication Mechanism

We propose a Multiple Event Channel Asynchronous Communication Mechanism(MECACM) to deal with the communication issue in the situations of multiple event channels coexistence. The architecture is depicted in Figure 1. It includes OKL4 hypervisor on top of hardware, a daemon server and two isolated VMs in user space which exist information exchange. The architecture contains the following two kinds of VMs:

Daemon server that runs in a separate cell (an architectural concept to describing the partitioning of system resources and communication channels between OKL4 programs including applications or VMs) to provide other cells with MECACM related services like multiple channel management, daemon thread information maintenance, waiting queue maintenance, IPC listener and message dispatcher. It encapsulates OKL4 underlying mechanisms and acts as the VM communication manager. For example, when a VM communicates with others using MECACM, it should firstly request daemon server to create an event channel. If the IPC listener catches the message, the daemon server triggers message dispatcher to deliver message to the receiver. If no receiver is ready, callers performing corresponding functions will wait in the queue and the message will be saved.

Communication cells that run Linux guest operating systems. Each communication cell starts a daemon thread in user space for event notification responses. The static nature of OKL4 requires that cells manually designate all of related runtime libraries and resource objects including threads, memory sections, etc. before compiling.

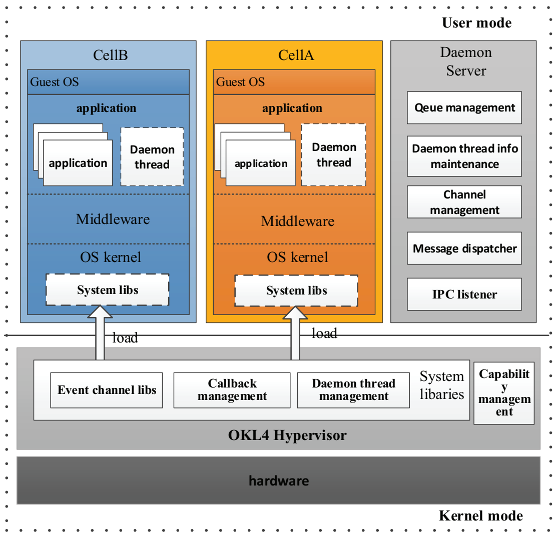

Figure.1: Architecture of multiple channel asynchronous communication mechanism

III. Evaluation

In this subsection we measure the performance of SMBDT mechanism and compare it with native OKL4FS mechanism. Similarly, we prepare a cell as daemon server and two cells as communication sides, but we also explore the influence shared memory size has on it. The comparison results are shown in figure 2.

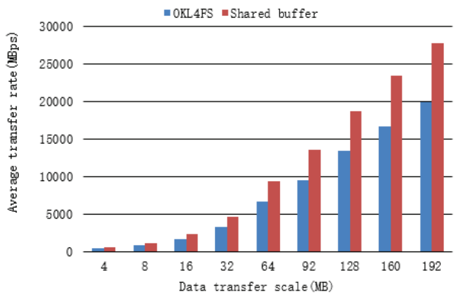

Figure.2: time overhead with 64KB shared memory

IV. Conclusion

A multiple event channel asynchronous communication mechanism based on OKL4 platform is proposed in this paper to resolve synchronous communication restrictions such as blocking and polling. The mechanism allocates specific event channel for each pair of communication entities to ensure the correctness of asynchronous mode and sets callback functions to automatically handle event notifications when triggered. In the end, the experiment results prove the feasibility and effectiveness of our mechanisms.

Acknowledgment

This work is supported by the Development Program of China (863 Program) No. 2015AA016105, National Key Research and Development Program of China No.2016YFB1000503, National High Technology Research, and National Natural Science Foundation of China No.61572062, No. 61732002, and No.61502019, and by Science and Technology on Communication Networks Laboratory Foundation under Grant No.KX132600025. The corresponding author of this paper is Yuebin Bai.

References

[1] Figueiredo R, Dinda P A, Fortes J. Guest Editors' Introduction: Resource Virtualization Renaissance. Computer, 2005, 38(5):28-31.

[2] Costa-Perez X, Swetina J, Guo T, et al. Radio Access Network Virtualization for Future Mobile Carrier Networks. IEEE Communications Magazine, 2013, 51(7):27-35.

[3] Aguiar A, Hessel F. Embedded systems' virtualization: The next challenge?// IEEE International Symposium on Rapid System Prototyping. 2010:1-7.

[4] Heiser G. Virtualizing embedded systems: why bother?// Design Automation Conference, DAC 2011, San Diego, California, Usa, June. 2011:901-905.

[5] Wessel S, Stumpf F, Herdt I, et al. Improving Mobile Device Security with Operating System-Level Virtualization. Computers & Security, 2015:148-161.

[6] Elphinstone K, Heiser G. From L3 to seL4 what have we learnt in 20 years of L4 microkernels?. Acm Sigops Symposium on Operating Systems Principles, 2013:133-150.

[7] Heiser G, Leslie B. The OKL4 microvisor: convergence point of microkernels and hypervisors. Journal of Urology, 2014:19-24.