A Survey of Virtual Machine System: Current Technology and Future Trends

Yunfa Li, Wanqing Li, Congfeng Jiang
Grid and Service Computing Lab, School of Computer Science and Technology
Hangzhou Dianzi University, 310018
Hangzhou, China
yunfali@mail.hust.edu.cn

Abstract— **With the development of the computer technology, the virtual machine has been become the main research topic. Understanding of the current technology and future trends of virtual machine system greatly help to improve the service performance of system. Therefore, we describe the current technology and present the future trends of virtual machine system in the paper. In the current technology of virtual machine system, we mainly describe the virtualization technology, the resource scheduling technology, the migration technology, the security technology and the performance evaluation technology. In the future trends of virtual machine system, we mainly present an overview of the future CPU architecture, the management mode of future memory and resource, the future maintaining method of system security and the performance evaluation method of future multiple virtual machine system.**

Keywords-virtual machine system; resource scheduling; migration; security; performance evaluation

I. INTRODUCTION

With the development of the computer technology, the virtual machine has been become the main research topic. By using the virtual technology, the computer system can aggregate all kinds of data resources, software resources and hardware resources and make these resources to provide service for different tasks. Moreover, the virtualization technology can separate hardware and software management and provide useful features including performance isolation [1], server consolidation and live migration [2]. In addition, the virtual technology can also provide portable environments for the modern computing systems [3]. Therefore, the new computing theorem and model that the virtualization technology embodies has very widespread use.

In generally, the architecture of virtual machine system can be shown in Figure 1. In the architecture, multiple virtual machines (VMs) share the same "physical machine", or host. At the lowest level, right above the hardware layer, the host OS kernel or virtual machine monitor (VMM) provides resource allocation to virtual machines. With each virtual machine, several tasks (or services) run on top of the "guest" OS which in turn provides the customary set of high-level abstractions such as file access and network support to applications running on the virtual machines. In fact, a virtual machine (VM) is a logical machine having almost the same architecture as a real host machine, running an operating system in it. A virtual machine system runs multiple virtual machines, each of which may run an operating system, in a single real host machine. Virtual machine allows users to create, copy, save (checkpoint), read and modify, share, migrate and roll back the execution state of machine with all the ease of manipulating a file. This flexibility provides significant value for users and administrators.

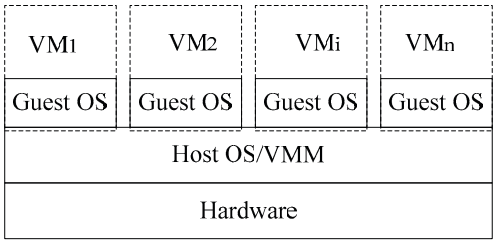

Figure 1. Architecture of virtual machine system

In a virtual machine system, the virtualization technology, the resource scheduling technology, the migration technology and the security technology play some key roles in determining the overall fairness and performance characteristics of the virtualized system. Traditionally, the virtual machine system has focused on fairly sharing the processor resources among domains. However, this can cause poor and/or unpredictable the quality of service of system. With the rapid growth of hardware and software resources, the performance evaluation of resource service in virtual machine system is becoming more and more important. Therefore, it becomes a key factor to improve the service performance of virtual machine system.

In this paper, we try to describe the current technology and present the future trends of virtual machine system. In the current technology of virtual machine system, we will mainly describe the virtualization technology, the resource scheduling technology, the migration technology, the security technology and the performance evaluation technology. In the future trends of virtual machine system, we will mainly present an overview of the future CPU architecture, the management mode of future memory and resource, the future maintaining method of system security and the performance evaluation method of future multiple virtual machine system.

The rest of this paper is organized as follows: Section 2 describes the current technology of virtual machine system.

978-0-7695-4219-5/10 $26.00 © 2010 IEEE
DOI 10.1109/ISECS.2010.80

Section 3 presents the future trends of virtual machine system. Finally, Section 4 presents some conclusion.

II. CURRENT TECHNOLOGY

Currently, the virtualization technology, the resource scheduling technology, the migration technology, the security technology and the performance evaluation technology have become the key technologies of virtual machine system. In order to understand the development state of virtual machine system, these key technologies are shown as follows, respectively.

A. Virtualization

Virtualization was first developed in 1960's by IBM Corporation, originally to partition large mainframe computer into several logical instances and to run on single physical mainframe hardware as the host. By day to day development, virtualization technology has rapidly attains popularity in computing. In fact, it is now proven to be a fundamental building block for today's computing.

The virtualization layer is the software responsible for hosting and managing all virtual machines on virtual machine monitor. Currently, virtualization approaches use either a hosted, or hypervisor architecture [4]. A hosted architecture installs and runs the virtualization layer as an application on top of an operating system, and supports the broadest range of hardware configurations. In contrast, hypervisor architecture installs the virtualization layer directly on a clean x86-based system. Depending on the needs and goals of the computer, some alternative techniques which provide for handling sensitive and privileged instructions to virtualize the physical resources are discussed as follows:

1) Full Virtualization: In this approach, kernel codes are translated to replace non-virtualizable instructions with new sequences of instructions that have the required effect on the virtual hardware. The guest OS is not aware it is being virtualized and requires no modification. The hypervisor simulates several logical instances of completely independent virtual computers possessing its own virtual resources. It translates all operating system instructions on the fly and caches the results for future use, while user level instructions run unmodified at native speed. The virtual resources included I/O ports and DMA channels. Therefore, each virtual machine can run any operating system supported by the underlying hardware. [5].

In Full Virtualization, the I/O devices are allotted to the guest machines by imitating the physical devices in the virtual machine monitor; interacting with these devices in the virtual environment are then directed to the real physical devices either by the host operating system driver or by the "hypervisor driver [5]". Therefore, Full virtualization can offer the best isolation and security for virtual machines; it simplifies migration and portability as the same guest OS instance can run on a virtualized or native hardware.

2) Paravirtualization: In this approach, the running guest OS should be modified in order to be operated in the virtual environment. Unlike full virtualization, Paravirtualization is a subset of server virtualization, which provides a thin software interface between the host hardware and the modified guest OS. Paravirtualization involves modifying the OS kernel to replace non-virtualizable instructions with hypercalls that communicate directly with the virtualization layer hypervisor. Moreover, the virtual machine monitor is simple which allows paravirtualization to achieve performance closer to nonvirtualized hardware.

Xen [4] is an example of Paravirtualization. It virtualizes the processor and memory using a modified Linux kernel and virtualizes the I/O using custom guest OS device drivers. Modifying the guest OS to enable Paravirtualization is relatively easy, compared to Full Virtualization.

3) Hardware Assisted Virtualization: Hardware assisted virtualization is a new CPU execution mode. In the new CPU execution mode, the VMM is allowed to run in root mode. Moreover, this mode allows privileged and sensitive calls to automatically trap to the hypervisor, removing the need for either binary translation or Paravirtualization. When Intel and AMD released their processors with inbuilt hardware which supports virtualization, the mode has recently gains attention. In addition, the hardware support virtualization architecture creates a trusted "root mode" and an untrusted "non-root mode". [6] in this mode. There is an example AMD-V [7] which supports hardware assisted virtualization.

4) Resource Virtualization: In a virtualizing system, "storage volumes, name spaces and the network resources" is regarded as resource virtualization [8]. There are various approaches to perform resource virtualization. For example, individual components may aggregated into a larger resource pool and a single resource such as disk space can partied into number of smaller and easily accessible resources of same type. In fact, storage virtualization is a form of resource virtualization, where a logical storage is created by abstracting all the physical storage resources that are scattered over the network. First the physical storage resources are aggregated to form a storage pool which then forms the logical storage. This logical storage which is the aggregation of scattered physical resources appears to be a single monolithic storage device to the user. The other resources virtualization is similar to the storage virtualization.

B. Migration

There are many reasons for the migration of a virtual machine, corresponding memory and file system. For example, from the point of view of a system administrator, the ability to migrate an entire virtual machine across hardware simplifies the issue of server maintenance. An operating system can be migrated by the administrator to a secondary and take the primary machine offline for servicing purposes, which contains a web server running on a primary machine. In order to improve the reliability of virtual machine system, the migration technology of virtual machine, the memory migration technology and the file system migration technology are presented. These technologies can be simply shown as follows.

1) Virtual Machine Migration: In this section, some different architecture will be briefly described, which have implemented virtual machine migration techniques. These technologies can be described as follows.

Xen is an x86 virtual machine monitor. In this virtual machine monitor, multiple commodity operating systems are allowed to share conventional hardware in a safe and resource managed fashion. The Xen hypervisor (the VMM) has direct access to the hardware, above which are the Xen domains (VMs) running guest OS instances. Each guest OS uses a pre-configured share of physical memory. A privileged domain called Domain0 (or Dom0), performs the tasks to create, terminate or migrate other guest VMs. Xen uses a send/recv model based on capabilities for transfer of states across VMs, implementing a standard "two-sided" interface.

Zap describes a novel system for transparent migration of applications [9]. It supports transparent migration of legacy and networked applications and provides a thin virtualization layer on top of the operating system that introduces a Process Domain (pod) abstraction. In this kind of system, each pod represents a process group with the same virtualized view of the system and a private namespace. This virtualized view associates virtual identifiers with OS resources such as PIDs and network addresses. This decouples processes in a pod from host dependencies, and forms the basic unit of migration.

In [10], the authors suggest a different approach to migrating virtual machines. Instead of traditional host driven migration, the authors propose self-migration of virtual machines and claim there are additional benefits when virtual machines are migrated in this manner. For example, there would be less overhead incurred from communication between the VMM and the virtual machine as well as increased security benefits. The network and CPU cost of performing the migration is attributed to the guest OS, rather than to the host environment. Portability is another benefit of self migration. Since migration happens without hypervisor involvement, this approach is less dependent on the semantics of the hypervisor and can be ported across different hypervisors and microkernels.

2) Memory Migration: In a virtual machine system, memory migration is one of the most important aspects of virtual machine migration. In general, the memory migration can be classified into three phases: namely push phase, stop-and-copy phase and pull phase. In the push phase, the source virtual machine continues running while certain pages are pushed across the network to the new destination. To ensure consistency, the pages modified during the transmission process must be re-sent. In the stop-and-copy phase, the source virtual machine is stopped, pages are copied across to the destination virtual machine, and then the new virtual machine is started. In the pull phase, the new virtual machine starts its execution, and if it accesses a page that has not yet been copied, this page is faulted in, across the network from the source virtual machine.

Practical solution paradigms include schemes incorporating mostly one or two of the above phases. For example, Internet suspend-resume technique uses pure stop-and-copy as its memory migration paradigm. In addition, this technique applies certain basic heuristics in order to reduce the content to be migrated. Similarly, pure demand-migration technique uses stop-and-copy to transfer essential kernel data structures to the destination, which is based on page faults at the site being transferred to. Pre-copy technique incorporates iterative push phases and a stop-and-copy phase which lasts for a very short duration [10]. In short, the pages to be transferred during round 'n' are only the ones dirtied during round 'n-1'. But there will always be a certain set of pages which are being updated so frequently that they can never be served by pure pre-copy iterations.

C. Resource Scheduling

In the virtual machine system, system resources are managed and controlled by a virtual machine monitor. Each virtual machine schedules the system resources for different tasks by using some resource scheduling algorithms, which are provided by the virtual machine monitor. These resource scheduling algorithms can be simply shown as follows.

The borrowed virtual time (BVT) scheduling algorithm is described in [11]. The essential of this algorithm is fair-share scheduler based on the concept of virtual time, dispatching the runnable virtual machine (VM) with the smallest virtual time first. Moreover, the algorithm provides low-latency support for real-time and interactive applications by allowing latency sensitive clients to "warp" back in virtual time to gain scheduling priority. The client effectively "borrows" virtual time from its future CPU allocation.

The Simple Earliest Deadline First (SEDF) scheduling algorithm is presented in [12]. In this algorithm, each domain specifies its CPU requirements. After all runnable domains receive their CPU share, SEDF will distribute this slack time fairly manner. In fact, the time granularity in the definition of the period impacts scheduler fairness.

The Credit Scheduling algorithm is described in [13]. It is Xen's latest proportional share scheduler featuring automatic load balancing of virtual CPUs across physical CPUs on an SMP host. Before a CPU goes idle, it will consider other CPUs in order to find any runnable virtual CPU (VCPU). This approach guarantees that no CPU idles when there is runnable work in the system.

In [14], the authors present a novel virtual I/O scheduler (VIOS) that provides absolute performance virtualization by being fair in sharing I/O system resources among operating systems and their applications, and provides performance isolation in the face of variations in the characteristics of I/O streams. In the scheduler, the VIOS controls the coarse-grain allocation of disk time to the different operating system instances and the output scheduler may determine the fine-grain interleaving of requests from the corresponding operating systems to the storage system.

D. Security

In a virtual machine system, the computer that is being virtualized is vulnerable to all the traditional attacks and exploits that are common to the normal environment. Therefore,

the security expectations are higher in the virtual machine system than that in normal environment. Moreover, there are possible points of entry, more holes to patch and more interconnection points in the virtual machine system. In order to ensure the security of virtual machine system, some security mechanisms and methods are presented. These security mechanisms and methods can be described as follows.

An experience of use of virtual machines for the security of systems was described in [15]. In the paper, Revirt is defined as an intermediate layer between the monitor and the host system, and the captured data is sent to the host system through the syslog process (the standard UNIX logging daemon) of the virtual machine. However, if the virtual system is compromised, the log messages can be manipulated by the invader and consequently are no more reliable. A VMI-IDS (Virtual Machine Introspection Intrusion Detection System) is described for searching intrusion evidences in [16]. In the system, the virtual machine executes directly on top of the hardware and the intrusion detection system executes in a privileged virtual machine and scans data extracted from the other VMs. The Secure Hypervisor (sHype) project [17] aims to support controlled sharing of resources between VMs on a platform, such as memory, CPU cycles, and network bandwidth. The above mentioned projects didn't consider the security of the VMM itself.

E. Performance Evaluation

With the development of virtual machine technology, the performance of virtual machine begins to be widely concerned. In order to research the performance of virtual machine, people present a lot of methods and make a great progress. These methods involve a lot of fields of virtual machine system. They can be simply shown as follows.

Menasc´e presents an analytic performance model for the general virtualized systems [18]. In the analytic performance model, the author uses an analytic queueing method to evaluate the performance of virtual environments. Bolker and Ding [19] discusses the analytic queuing models for the virtualized system. They focused on the models on processor utilization and finally tested their model to study the VMware by a benchmark. In [20], Menon et al present a diagnosing performance overhead method about resource scheduling in the xen virtual machine environment. In this method, a toolkit is used to analyze performance overheads incurred by networking applications running in Xen VMs. The toolkit enables coordinated profiling of multiple VMs in a system to obtain the distribution of hardware events such as clock cycles and cache and TLB misses. In [21], the authors analyze and compare the CPU schedulers in the Xen virtual machine monitor (VMM) in the context of traditional workload managers. They use the open source Xen virtual machine monitor to perform a comparative evaluation of three different CPU schedulers for virtual machines and analyze the impact of the CPU scheduler and resource allocation on application performance. In [22], the monitoring method is used to evaluate the performance of a virtualization system. In the monitoring method, xm and Xenoprof are powerful tools to realize system profiling. In [23], Baba et al. proposes a disk access throughput evaluation method in virtual machine environments where multiple independent virtual machines share a common physical shared disk drive. In [24], Ye et al. provide a framework to analyze the performance of virtual machines system, which is based on the queueing network models. In the framework, the virtual machines either do not run at all or just monitor the virtual machines instead of the hypervisor.

III. FUTURE TRENDS

Owing to most modern CPU architecture were not designed to be virtualizable, the development of virtual machine technology is very slowly. Only a new execution mode is added to the processor, Intel with its Vanderpool technology and AMD with its Pacifica technology can support for x86 CPU virtual machine monitors. The new processor can let a VMM safely and transparently use direct execution for running virtual machines. In order to improve performance, it is necessary for the future mode to reduce both the traps needed to implement virtual machines and the time it takes to perform the traps. Only these technologies become available, direct-execution-only virtual machine monitors could be possible on x86 processors.

Resource management holds great promise as an area for future research. Much work remains in investigating ways for virtual machine system and guest operating systems to make cooperative resource management decisions. In addition, research must look at resource management at the entire data center level, and we expect significant strides will be made in this area in the coming decade.

With the rapid growth of hardware and software resources, the management about virtual machine system is becoming more and more difficult. Moreover, the question that people needs to solve is becoming more and more complicated. Moreover, the utilization ration and the service performance of resource will reduce with the growth of the scale of computer system. Thus, there are two inconsistent factors between how to expand the scale of the computer system and how to improve the utilization ration and the real-time performance of resource service. In order to schedule the system resource and efficient improve the service performance of resource, it is necessary for the future mode to built a real time interactive strategy for the multiple virtual machine system in different virtual machine monitors.

In a virtual machine system, the security of system main involves the security of the host and the hypervisor. If the host or the hypervisor is compromised then the whole security model is broken. Attacks against the hypervisor will become more popular among the attackers realm. Therefore after setting up the environment, it should be taken to ensure that the hypervisor is secure enough to the newly emerging threats, if not patches has to be done. Patches should be done frequently so that the risk of hypervisor being compromised will be avoided.

Although current some performance monitoring methods can monitor or predict the performance of virtual machine, they will confront a lot of difficulties in multiple virtual machine system because there are some different in the virtual machine system and the multiple virtual machine system, such as the state of resource scheduling and the processes of task

processing. In order to overcome these disadvantages, some new performance evaluation model, performance monitoring methods should be developed for the multiple virtual machine system.

IV. CONCLUSION

The paper describes the current technology and presents the future trends of virtual machine system. In the current technology of virtual machine system, we mainly describe the virtualization technology, the resource scheduling technology, the migration technology, the security technology and the performance evaluation technology. In the future trends of virtual machine system, we mainly present an overview of the future CPU architecture, the management mode of future memory and resource, the future maintaining method of system security and the performance evaluation method of future multiple virtual machine system.

ACKNOWLEDGMENT

This paper is supported by Zhejiang Povincial Natural Science Foundation of China under Grant No. Y1090297 and Y6090312, Applied Research Program of Nonprofit Technology of Zhejiang Province under Grant 2010C31G2040121, Startup Foundation of School under grant No. KYS055608103, and Open Foundation of Services Computing Technology and System Lab

REFERENCES

[1] D. Gupta, L. Cherkasova, R. Gardner, and A. Vahdat. “Enforcing Performance Isolation across Virtual Machines in Xen”. In Proceedings of the 7th International Middleware Conference, LNCS Press, 2006. pp.342-362

[2] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt and A. Warfield. “Live Migration of Virtual Machines”. Proceedings of the 2nd Symposium on Networked Systems Design and Implementation, Boston, Massachusetts, USA, May 2005.

[3] K. Fraser, S. Hand, R. Neugebauer, I. Pratt, A. Warfield, and M. Williamson. “Safe Hardware Access with the Xen Virtual Machine Monitor”. Proceedings of the 1st Workshop on Operating System and Architectural Support for the on demand IT InfraStructure (OASIS), Boston, MA, October 2004.

[4] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, R. N. Alex Ho, I. Pratt, and A. Warfield, “Xen and the Art of Virtualization”. Proceedings of the 19th ACM Symposium on Operating Systems Principles, ACM Press, October 2003, Bolton Landing, NY, USA, pp.164-177

[5] W. Huang, Q. Gao, J. Liu, and D. K. Panda, “High Performance Virtual Machine Migration with RDMA over Modern Interconnects”, Proceedings of IEEE Conference on Cluster Computing (Cluster 2007), Austin, Texas. September 2007, pp.11-20, doi: 10.1109/CLUSTR.2007.4629212

[6] P. Apparao, S. Makineni, D. Newell, “Characterization of Network Processing Overheads in Xen”. First International Workshop on Virtualization Technology in Distributed Computing, Nov. 2006, pp.2-2

[7] G. Vallee, T. Naughton, C. Engelmann, O. Hong, S. L. Scott, "System-Level Virtualization for High Performance Computing". Proceeding of the 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing, Feb. 2008, pp.636-643, doi: 10.1109/PDP.2008.85

[8] C. H. Huang, P. A. Hsiung, "Hardware Resource Virtualization for Dynamically Partially Reconfigurable Systems". IEEE Embedded Systems Letters, May 2009, Vol. 1, No. 1, pp.19-23, doi: 10.1109/LES.2009.2028039

[9] S. Osman, D. Subhraveti, G. Su, and J. Nieh, The Design and Implementation of Zap: A System for Migrating Computing Environments, 5th Symposium on Operating Systems Design and Implementation (OSDI 2002), Boston, MA, December 2002

[10] J. G. Hansen, and E. Jul, “Self-migration of Operating Systems”, Proceedings of the 11st ACM SIGOPS European Workshop, Leuven, Belgium, September 2004, pp.23

[11] B. Lin, P. A. Dinda. “Towards Scheduling Virtual Machines Based On Direct User Input”. First International Workshop on Virtualization Technology in Distributed Computing, IEEE Press, November 2006, pp.6-6, doi: 10.1109/VTDC.2006.15

[12] K. Korotaev, “Hierarchical CPU Schedulers for Multiprocessor Systems, Fair CPU Scheduling and Processes Isolation”, IEEE International Conference on Cluster Computing, IEEE Press, September 2005, pp.1-1

[13] J. F. An, X. Y. Fan, S. B. Zhang, et al. “An Efficient Verification Method for Microprocessors Based on the Virtual Machine”, First International Conference on Embedded Software and Systems, LNCS Press, August 2005, Vol. 3605/2005, pp.514-521

[14] S. R. Seelam, P. J. Teller, “Virtual I/O Scheduler: a Scheduler of Schedulers for Performance Virtualization”. Proceedings of the 3rd International Conference on Virtual Execution Environments (VEE 2007), San Diego, California, USA, ACM press, June 2007, pp.105-115, doi: 10.1145/1254810.1254826

[15] G. Dunlap, S. King, S. Cinar, M. Basrai, P. Chen, “ReVirt: Enabling Intrusion Analysis through Virtual-Machine Logging and Replay”. Proceedings of the Operating Systems Design and Implementation (OSDI), Boston, Massachusetts, USA, Dec. 2002.

[16] T. Garfinkel, M. Rosenblum, “A Virtual Machine Introspection Based Architecture for Intrusion Detection”. Proceedings of the Network and Distributed System Security Symposium (NDSS), 2003.

[17] R. Sailer, T. Jaeger, E. Valdez, R. Caceres, R. Perez, S. Berger, J. L. Griffin, L. Doorn, “Building a Mac-based Security Architecture for the Xen Open-source Hypervisor”. Proceeding of the 2005 Annual Computer Security Applications Conference, Dec. 2005, pp. 276–285, doi: 10.1109/CSAC.2005.13

[18] D. A. Menasc´e, “Virtualization: Concepts, Applications, and Performance Modeling”, Proc. 31th Int. Computer Measurement Group Conf, 2005, pp. 407–414,

[19] E. Bolker and Y. Ding, “Virtual Performance won’t Do Capacity Planning for Virtual Systems”, Proc. 31th Int. Computer Measurement Group Conf, 2005, pp. 1-39,

[20] A. Menon, J. R. Santos, Y. Turner, G. J. Janakiraman, and W. Zwaenepoel, “Diagnosing Performance Overheads in the Xen Virtual Machine Environment”, Proceedings of the 1st International Conference on Virtual Execution Environments (VEE 2005), June 11-12, 2005, Chicago, IL, USA, pp.13-23, doi: 10.1145/1064979.1064984

[21] L. Cherkasova, D. Gupta, and A. Vahdat. “Comparison of the Three CPU Schedulers in Xen”. SIGMETRICS Performance Evaluation Review, 2007, 25(2), pp.42-51

[22] P. Willmann, J. Shafer, D. Carr, etc, “Concurrent Direct Network Access for Virtual Machine Monitors”, Proceedings of the IEEE 13th International Symposium on High Performance Computer Architecture, IEEE Press, February 2007, pp.306-317, doi: 10.1109/HPCA.2007.346208

[23] T. Baba, and A. Tanaka, “Simple and Practical Disk Performance Evaluation Method in Virtual Machine Environments”, International Symposium on Performance Evaluation of Computer and Telecommunication Systems, 2008. SPECTS 2008. June 2008, pp.338-345

[24] D. S. Ye, Q. M. He, H. Chen, and J. H. Che, “A Framework to Evaluate and Predict Performances in Virtual Machines Environment”, IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, 2008. EUC '08. Dec. 2008, Vol. 2, pp.375-380, doi: 10.1109/EUC.2008.42