Virtual Environments - Framework for Virtualized Resource Access in the Grid

Michał Jankowski[1], Paweł Wolniewicz[1], Jiří Denemark[2], Norbert Meyer[2], and Luděk Matyska[23]

[1] Poznań Supercomputing and Networking Center,ul. Noskowskiego 10, 61-704 Poznań, Poland
{jankowsk,meyer,pawelw}@man.poznan.pl
[2] Faculty of Informatics, Masaryk University, Botanická 68a, 602 00 Brno, Czech Republic
jirka@ics.muni.cz
[3] Institute of Computer Science, Masaryk University, Botanická 68a, 602 00 Brno, Czech Republic
ludek@ics.muni.cz

Abstract. To assure secure access to any computer resources one must provide an adequate level of authentication, authorization job isolation and possibility of auditing user actions. In the grid environment that comprises a large number of users and resources in different administrative domains, these features are challenging. Grid economy and accounting related to it are becoming more and more important in an emerging aspect of grid commercialization. Also, the requirements of the users and administrators are becoming more and more sophisticated: checkpointing and migration of jobs, detailed software requirements, quality of service, collaborative work, and load balancing, to name a few. Virtualization techniques, nowadays more and more matured and advanced, seem to help solve the above-mentioned problems. In the present paper we discuss some of these techniques as well as existing solutions and then propose a framework for Virtual Environments. The framework focuses on resource access control, but the benefits of virtualization are wider.

1 Introduction

Controlled and secure access to grid computational resources requires authentication, authorization, an adequate level of job isolation and possibility of auditing user actions. This should be realized with as little administrative effort as possible, though providing the administrators and Virtual Organization (VO) managers with enough control on their resources and users. Grid economy, which introduces accounting and billing requirements, also becomes more and more important. From the users point of view the whole Grid should be seen as a single computer with appropriate software, hiding all the technical details connected with physical locations, middleware, operating systems, etc. The mentioned groups of requirements (described in detail in [13,14]) are closely related on the conceptual level.

W. Lehner et al. (Eds.): Euro-Par 2006 Workshops, LNCS 4375, pp. 101–111, 2007.

We have researched a number of existing solutions and found that there are tools that provide for at least part of the functionality we are interested in, however none of them addresses all the issues. These tools are widely used in numerous projects and some of them have become standard, so it seems to be reasonable to be compatible with them. Moreover, different users, resource owners and VO managers may have different and often conflicting needs. For example, there is no isolation level that would always be suitable; sometimes complete encapsulation of jobs is a must, sometimes jobs need collaboration or strict isolation is too heavy solution. Hence we have several components ("building blocks") that could be used to build a perfect solution for a given situation, but we need a way to put them together into a framework that will combine these tools and their features gaining the synergy effect.

Virtualization technology has a long history in computer science [1]. It allows for partitioning or combining real components of computer infrastructure (hardware, software, networking, etc.) into virtual entities. This technique abstracts from internal details of physical elements, provides isolation and common interface for virtual elements, even if they share physical entities. Examples are virtual memory, virtual machines and virtual networks. This technology seems to be especially promising for grid middleware as it must cover large, loosely coupled and heterogenous distributed systems and should hide its complexity from the user.

In our previous papers we have introduced a concept of the *Virtual Environment*, which we understand as encapsulation of user jobs in order to give a limited set of privileges and be able to identify the user and organization on behalf of which the job acts. Depending on requirements we may virtualize user accounts [10],[11] or virtual machines [7]. The concept of "combining real components" opens a way for dynamic construction of an environment. The virtualization technique simplifies the assignment of jobs to resources either by discovering a statically created environment or by expressing parameters of a dynamically created one, because the user's requirements specified in an abstract language may specify abstract features of the environment. In this work we describe an idea of a framework for the creation and managing of Virtual Environments. The structure of the paper is as follows: in section 2 we discuss advantages and disadvantages of Virtual Accounts (VA) and Virtual Machines (VM); section 3 describes the concept and implementation of Virtual Workspaces (VW), which we found especially useful in constructing our framework; section 4 provides architecture of our framework; accounting issues will be discussed in section 5, and finally section 6 concludes the paper.

2 Comparison of VA and VM

Both methods of virtualization: Virtual Accounts and Virtual Machines allow for running jobs in separated Virtual Workspaces, but they are best suitable for different purposes. Virtual Accounts is just a simple implementation of assigning users to different Unix accounts. Different directories and Unix accounts are used

to separate jobs. In case of workflow, tasks are run in the same account. Complete knowledge about mapping from real to the Virtual Account is stored locally and can be used to resolve all Unix accounts from standard accounting procedures. Virtual Machines have far more possibilities. In fact Virtual Machines run several instances of the operating system at the same time and thus provide complete job separation. Virtual Machines are best suitable for resource centers where job requirements differ, e.g. operating system requirements are different or even the grid infrastructure for different users group is incompatible.

Virtual Machines can be used in two ways. One way is to set up static Virtual Workspaces, for example, to run two different grid infrastructures, or to run different grid testbeds for different VOs. Virtual Machines (or rather Virtual Clusters) can also be set up on demand, for the lifetime of the job. This, however, causes some overhead because either the Virtual Machine must be created and started (which is time-consuming) or the Virtual Machine was created before and must now be resumed and reconfigured (which is memory-consuming).

Accounting is very important for the system administrator. The resources used must be calculated and stored. Standard accounting stores all information locally. This causes problems for Virtual Machines, because when the machine is deleted, all detailed information is lost. On the other hand, when the ma chine is migrated, the information may be inaccurate. Estimated accounting is still available from the Virtual Machine Management System (e.g. Xen[22,23]), but they combine all details into one set of numbers. For instance, it is not possible to distinguish between user time and system time, and information about executed command names are not available. The solution is to use an external database and send all information there during shutdown.

A similar limitation is connected with audit. Logging the operations performed locally on the VM, on its virtual resources is usually not interesting for the physical machine administrator, but access to some physical devices (e.g. laboratory equipment) or network connections may be the subject for audit. However, this may be difficult as some relevant logs may be located on VM and lost on deletion. Also if the VM user has root privileges, he may maliciously or accidentally modify or remove the logs.

Using Virtual Machines it is possible to provide a service level agreement (SLA). Resources assigned to the given Virtual Machine can be managed easily. SLA for systems with Virtual Accounts is limited. To some extent it can be achieved by careful configuration of the operating system and the queuing system.

The integration of virtual environments with the grid infrastructure is especially important. The Virtual Accounts can be easily integrated with Globus [15] or gLite [17], because it is just a plugin to the grid middleware. With Virtual Machines things are more complicated. Dynamic creation of Virtual Machines is not compatible with existing grid resource brokers. Resource broker does not know about wirtual environment, therefore it can not create wirtual workspace. The resource broker just contacts the head node and submits the job to the cluster. But in case of the Virtual Cluster the head node is not created yet and it should be created after the resource broker submits the job to the site. Therefore

Table 1. Summary of Virtual Accounts and Virtual Machines

	Virtual Accounts	Virtual Machines
Purpose	small clusters simple needs	Many VOs, many OS-es, Many jobs at a time, SLA
Flexibility	in some extent	very flexible
Job separation	limited	full
Accounting	full	limited
Audit	full trusty	limited may be untrusty
Administration	easy	difficult
SLA	limited	yes
integration with grid systems	easy	difficult
resource consumption	insignificant	small to large

the submission process must have two steps, and an additional module (GRAM Proxy) is needed. First, GRAM Proxy accepts the job from the resource broker and then creates the Virtual Machine and submits jobs there

For Virtual Machines there is also a problem of administration. Virtual Machines are set up from partitions stored somewhere on a hard disk. But Virtual Machines restored from the partition must be up to date, which means that after startup some configuration must be updated, e.g. gridmap files, certificate revocation list, some security patches etc. This causes additional time overhead and delays job starting.

Summary of features for virtual systems is presented in table 1.

In general, Virtual Machines have a big potential, but quite often all site requirement can be fulfilled with Virtual Accounts. For sites with thin nodes (single or dual processors) a typical configuration of job management systems allows for running only one job per node. In this way jobs are completely separated. For large nodes with many jobs running at the same time, dynamic assignment can be a very good solution, especially in the context of a service level agreement.

An intermediate solution with Virtual Machines set up statically to share the same hardware between different grid infrastructures and with Virtual Account used to ensure virtualization inside Virtual Machines is also possible.

3 Virtual Workspaces Approach

An architecture called virtual workspaces [3,4,5,6] has been designed to automate the creation and management of distributed dynamic virtual environments in the Grid. The architecture comprises several services used to create and manage virtual environments. When users want to submit a job to a Grid resource, they contact an appropriate service to create a dynamic virtual environment for them. For existing environment users can use another service to manage the environment, e. g., to change the environment's lifetime, to configure or terminate the environment. During the lifetime of the virtual environment, standard Grid services such as GRAM can be used for submitting jobs.

The Virtual Workspaces architecture does not enforce any virtual environment implementation. Currently, the implementation of Virtual Workspaces based on Dynamic Accounts and Virtual Machines is available. The background technology is not pluggable—it is chosen at install time and then only the selected implementation is available.

Virtual Workspaces can be simple (or atomic), and jobs are submitted directly into it, or it may consist of several (either atomic or complex) workspaces. Complex workspaces are used to create virtual clusters with a set of definitions of Virtual Workspaces for both the head node and worker nodes of a real cluster. According to a specification of a virtual cluster, several Virtual Machines are deployed on physical cluster nodes and set up to form an isolated private IP network. The virtual machine running on a head node is the only part of a virtual cluster with a public IP address. After all virtual machines forming the virtual cluster are up and running, a GRAM service is started on the virtual head node. Clients then use this GRAM service to start their jobs on the virtual cluster.

To support Virtual Workspaces, each node of a physical cluster must run the Xen Virtual Machine Monitor and several services for staging, starting and managing Virtual Machines.

As the Virtual Workspace is just an environment for submitting users' jobs, it must be accessible from everywhere for users to be able to contact its services. In other words, each Virtual Workspace (except for worker nodes of a virtual cluster) has to be provided with a public network address. This may cause problems especially when more than one Virtual Workspace is allowed to be created on a single physical machine.

On the other hand, users are provided with a way of deploying their own environment which perfectly suits their needs. However, if users or VO administrators are allowed to provide a complete image of a Virtual Machine, it must be done in such a way that site administrators are willing to trust the image.

If more detailed information on using specific resources is needed for accurate accounting, coarse runtime data obtained from the Virtual Machine Monitor may not be enough, and special monitoring tools providing data from the inside of a Virtual Machine have to be deployed. Similar tools might be useful for logging user activities.

4 Architecture of the Framework

In our previous papers [13,14] we described a set of different requirements for user management and access to resources. We stated that there are numerous tools that provide at least part of the required functionality, however none of them addresses all the issues. These tools are used in working Grids. We proposed to put them into a pluggable framework that will combine the features gaining the synergy effect.

Section 3 has described Virtual Workspaces effort in detail. Conceptually the Virtual Workspace is the same as the Virtual Environment defined in our previous papers. Moreover, VW implementation seems to fulfill most of our requirements and its architecture is quite similar to our framework. Similarly to our proposition VW employs WS–Stateful Resource [9] for modeling of the workspace and managing its life cycle. Hence we would like our framework implementation to be based on VW. In this section we will discuss how the VW fits our framework, and which elements should be added or modified.

As we discussed in the previous section, both Virtual Environment implementations (Dynamic Accounts aka Virtual Accounts and Virtual Machines) have their pros and cons. The decision on using or not VE and which one is the preferable implementation is up to the resource administrator. This fact should be transparent from the user point of view, but the VW require explicit create and life time management operations and to make things worse, these two implementations provide slightly different interfaces. As a result, the party that requests the job run (either a resource broker or directly the user - let's call them both "client") must take care of workspace management and be conscious of actual interface. These operations may be necessary for advanced global schedulers that support SLA or checkpointing and workspace migration. Explicit workspace management is still redundant from the point of view of the clients in most cases. The user just wants to run a job with specified parameters and creation of the workspace is a technical detail that should be hidden. Also most of existing brokers would require modification in order to support the VW operations. The appreciated scenario is that the client calls the resource manager service (like Globus GRAM) directly and without a previous request for the workspace creation.

We propose to hide the creation and lifetime management inside the resource manager, that will take care of the creation automatically. Any special user requirements concerning hardware (number of nodes, memory, etc.), operating system and software may be expressed in the job description. These information passed to the resource manager may be used for the Virtual Environment creation. Any creation parameters, that are not explicitly specified may obtain default values. The Virtual Environment must live until the job is finished at least, then it may be destroyed. The destruction may be performed periodically or when the resources occupied by the VE are needed by someone else.

The described architecture is shown on figure 1. The newly proposed parts are modified Globus GRAM (may be both WS and pre-WS one) which accepts job management requests, VE database and VE Create & Mapping module that

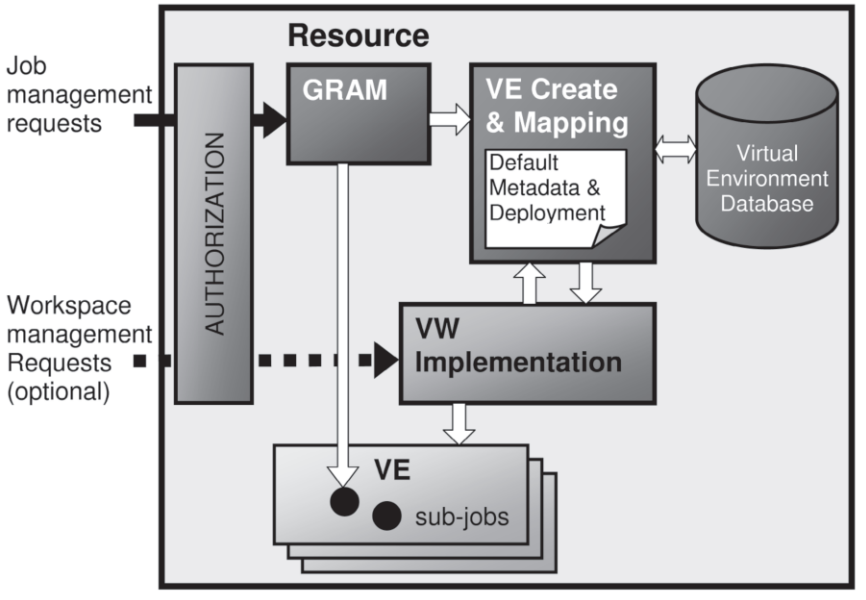

Fig. 1. Architecture of the Framework

interfaces the GRAM with VE database and VW implementation. Webservice interface of VW may be accessible outside optionally and if this is the case, VW operations must be synchronized with the Create & Mapping.

One of the most important features connected with the resource management is fine grained and flexible authorization. The GT 4.0 Authorization Framework [18,19] allows for a variety of authorization schemes, including a gridmap-file, an access control list defined by a service, an SAML-based authorization service and any custom authorization handler. The security descriptors allow for flexible security configuration on different levels: container, service, and even resource. There is a number of existing authorization systems and mechanisms that already are or easily may be plugged into this Globus framework and fulfill our authorization requirements. The administrator may properly configure the Virtual Workspaces and WS GRAM services according to the local needs. The pre–WS authorization is not equally flexible, but it is still possible to implement its own, fine grained authorization using callouts mechanisms [20,21].

Note, that the authorization is closelly related to the workspace creation and mapping user to the workspace. The limitations put on the workspace (e.g. privileges of the virtual account or resources allocated to virtual machine) are simply security enforcement mechanisms. Moreover, the following job run or file transfer requests with the same credencials should be mapped to the same environment.

In case of VA implementation, creation of the environment is virtually equivalent to the mapping operation and may be easily realized by the GRAM mapping module. The environment is simply a record in the VE database that binds user

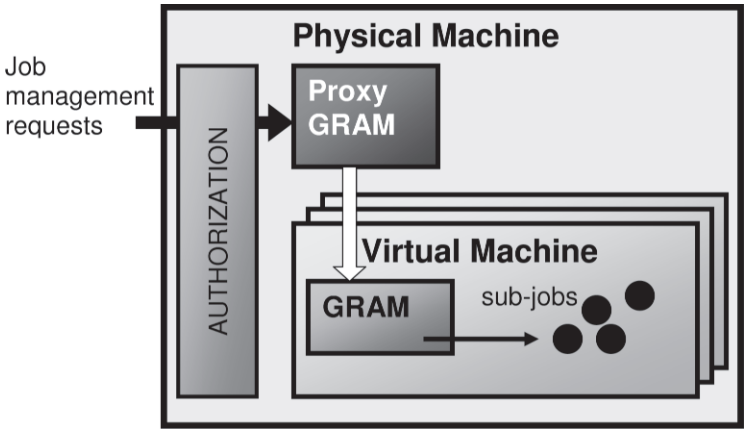

Fig. 2. Proxy GRAM

to a virtual account. Note that VM meta data and deployment parameters are equivalents of the static parameters of a physical machine with the VA system. In case of VA the parameters are only evaluated if they fit to the real resources (e.g. if required software is installed).

In case of VM implementation, virtual cluster matching the user requirements must be actually created by the resource manager. The resource manager, that actually runs the job, is located on the head node of the virtual cluster. In order to make this fact invisible for the client, all client requests are accepted by the GRAM that is located on the physical machine (e.g. on domain0 of Xen). It is called "proxy GAM" in that case. The proxy will access the VE database in order to set/get the current user – VE mapping, create the VM if necessary and forward the job request to the "internal" GRAM see figure 2.

The Virtual Workspaces implementations are missing a database that might be used for storing history of mappings user – Virtual Environment which is crucial for accounting and auditing purposes. The following section describes in more detail what should be stored in this database, how these data might be obtained from the underlying system and how the information would be exposed outside.

5 Accounting and Audit

The auditing or accounting data is normally bound to a local account or Virtual Machine instance. However, in a grid system one is interested in this information in the context of the global user identity and his Virtual Organization. The records of VE operations together with the standard system logs and accounting data provide complete information on user actions and resource usage, but these two sources must be combined. Virtual Environment Information Subsystem enables this feature. It consists of VE Database, VE Information Service and framework for collecting accounting and audit events - see figure 3.

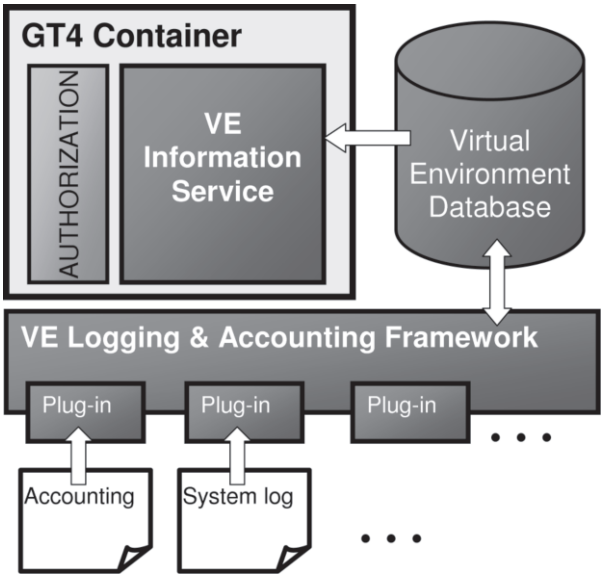

Fig. 3. Architecture of Virtual Environment Information Subsystem

The Virtual Environment Database stores all the relevant information connected with the VE creation and deletion, just on the request of the VE services. The database is also capable of storing any type of accounting data, both standard and nonstandard, and any kinds of events described in string values, all of this unified and connected to the grid user. These data may be collected periodically or on request (e.g. just after the VE is deleted), by analyzing sources like Unix accounting records, system logs etc. The sources may be quite different, depending on VE implementation, operating system, used software etc. so we use a pluggable framework. A plugin must be implemented for each source.

The Virtual Environment Information Service is a frontend for the Virtual Environment Database. Access to the data must be authorized and depends on the users role: all the users have rights to read the accounting data referring to themselves, managers of virtual organizations are able to read data referring to all VO members, owners of resources are allowed to read all the data connected to the resource.

As stated in section 2, the Virtual Machine implementation introduces some problems connected with the accounting and audit data gathering. This may be overcome by careful configuration and running some software on VM that will put the relevant information to the VE database on the physical machine. This workaround, however, will result in lower flexibility of the machine (e.g. the VM is not fully transparent for the migration process).

6 Conclusions

In the paper we have shown how the virtualization techniques simplify access and administration of grid resources, and which solutions may be useful depending on the situation. We have discussed the leading solution in the area: Virtual Workspaces. We have also proposed a framework based on VW which allows for easy integration of numerous existing grid middleware components. VW Our contribution to VW is as follows: automatic (transparent for the client: user or resource broker) creation of the virtual environment, database and service supporting accounting and audit features.

Acknowledgment

This work has been supported by the CESNET Research Intent (MSM6383917201) and by the EU CoreGRID NoE (FP6-004265). Implementation of Virtual Accounts System is included in EU BalticGrid project (RI-026715) and in Clusterix - National Cluster of Linux Systems, project co-funded by the Polish Ministry of Sciences.

References

1. A.Singh: An Introduction to Virtualization. http://www.kernelthread.com/publications/virtualization/ (2004)
2. I.Foster, C.Kesselman, S.Tuecke: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International J. Supercomputer Applications **15(3)** (2001)
3. I.Foster, T. Freeman, K.Keahey, D.Scheftner, B.Sotomayor, X.Zhang: Virtual Clusters for Grid Communities. CCGRID 2006, Singapore (2006).
4. K.Keahey, I. Foster, T. Freeman, X. Zhang: Virtual Workspaces: Achieving Quality of Service and Quality of Life in the Grid. Scientific Programming Journal, Volume 13, **4/2005** (2005)
5. X.Zhang, K.Keahey, I.Foster, T.Freeman: Virtual Cluster Workspaces for Grid Applications. ANL/MCS-P1246-0405 (April 2005)
6. K.Keahey, I.Foster, T.Freeman, X.Zhang, D.Galron: Virtual Workspaces in the Grid. Europar 2005, Lisbon, Portugal (September 2005)
7. K.Keahey, K Doering, I.Foster: From Sandbox to Playground: Dynamic Virtual Environments in the Grid. 5th International Workshop in Grid Computing (Grid 2004), Pittsburgh, PA (November 2004)
8. K.Keahey, M.Ripeanu, K.Doering: Dynamic Creation and Management of Runtime Environments in the Grid. Workshop on Designing and Building Web Services (GGF 9), Chicago, IL (October, 2003)
9. I.Foster, J.Frey, S.Graham, S.Tuecke, K.Czajkowski, D.Ferguson, F.Leymann, M.Nally, I.Sedukhin, D.Snelling, T.Storey, W.Vambenepe, S.Weerawarana: Modeling Stateful Resources with Web Services, version 1.1. http://www-128.ibm.com/developerworks/library/specification/ws-resource/ (March 2004)
10. M.Kupczyk, M.Lawenda, N.Meyer, P.Wolniewicz: Using Virtual User Account System for Managing Users Account in Polish National Cluster. HPCN, Amsterdam, (June 2001)

11. M.Jankowski, P.Wolniewicz, N.Meyer: Virtual User System for Globus based grids. Cracow '04 Grid Workshop, (December 2004)
12. J.Denemark, M.Jankowski, A.Krenek, L.Matyska, N.Meyer, M.Ruda, P.Wolniewicz: Best Practices of User Account Management with Virtual Organization Based Access to Grid. 6th International Conference, PPAM 2005, Springer-Verlag LNCS 3911, Poznan, (September 2005)
13. J.Denemark, M.Jankowski, L.Matyska, N.Meyer, M.Ruda, P.Wolniewicz: User Management for Virtual Organizations. CoreGRID Integration Workshop, Pisa (2005)
14. J.Denemark, M.Jankowski, L.Matyska, N.Meyer, M.Ruda, P.Wolniewicz: CoreGRID Technical Report TR-0012: User Management for Virtual Organizations. (2005)
15. http://www.globus.org
16. http://workspace.globus.org
17. http://glite.web.cern.ch/glite/
18. http://www.globus.org/toolkit/docs/4.0/security/authzframe/
19. B.Lang, I.Foster, F.Siebenlist, R.Ananthakrishnan, T.Freeman: A Multipolicy Authorization Framework for Grid Security. Accepted by the IEEE NCA06 Workshop on Adaptive Grid Computing (to appear in Proc. Fifth IEEE Symposium on Network Computing and Application), Cambridge, USA (July 2006)
20. http://www.globus.org/toolkit/security/callouts/
21. GSI Admission Control and Identity Mapping Callout Specification, Draft. (July 1, 2003)
22. P.Barcham at al: Xen 2002. University of Cambridge Computer Labolatory Technical Report UCAM-CL-TR-553, Cambridge, (January 2003).
23. P.Barham, B.Dragovic, K.Fraser, S.Hand, T.Harris, A.Ho, R.Neugebauer, I.Pratt, A.Warfield: Xen and the Art of Virtualization. Symposium on Operating Systems Principles (SOSP '03) (October 2003)